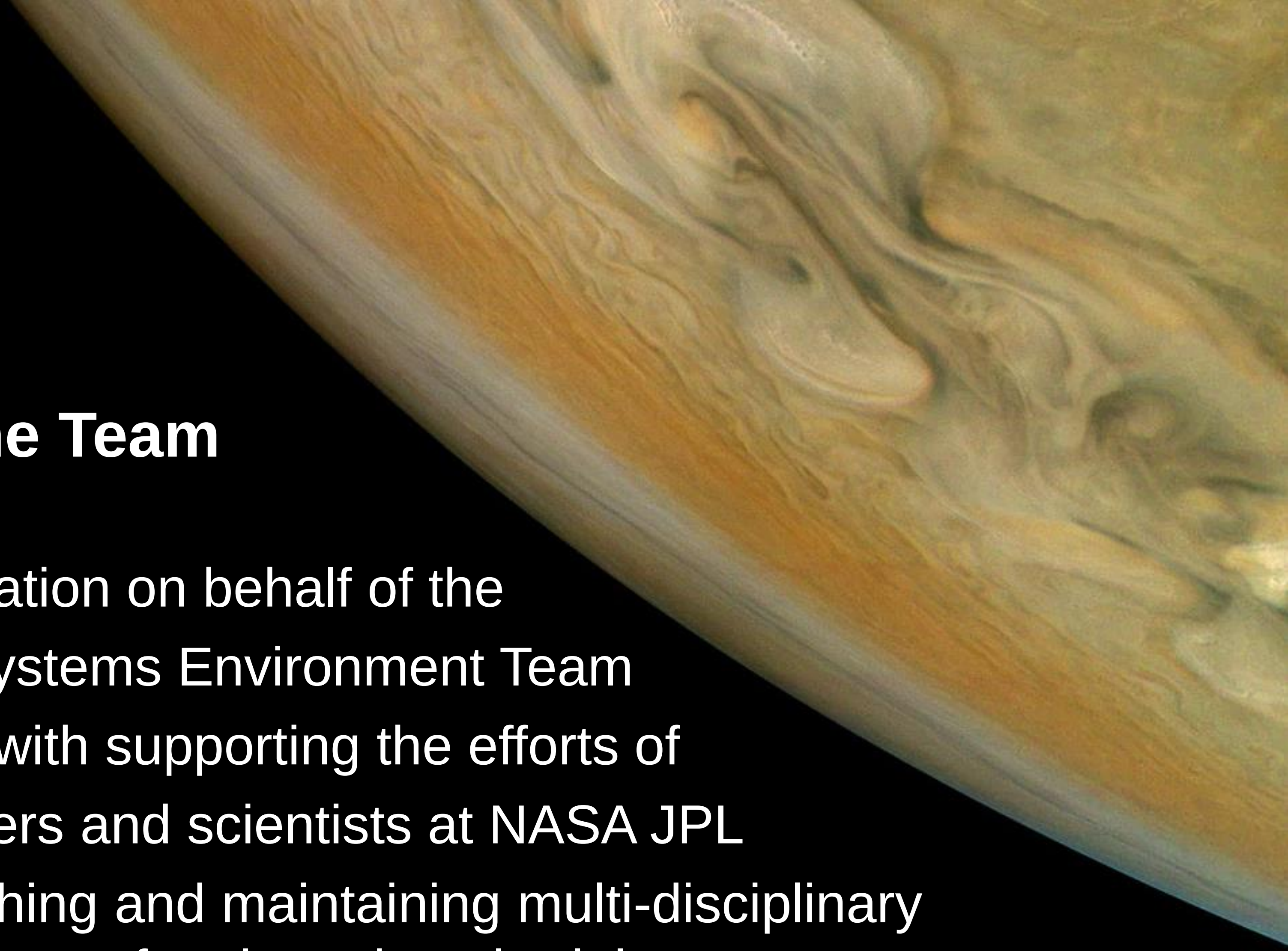
NASA JPL Systems Environment

Jet Propulsion Laboratory, California Institute of Technology

Eric W Brower

17-19 April 2018 – Phoenix International Users' Conference, Annapolis MD, USA



About the Team

- Presentation on behalf of the CAE Systems Environment Team
- Tasked with supporting the efforts of engineers and scientists at NASA JPL
- Establishing and maintaining multi-disciplinary integrations of tools and methodology

About the Presenter

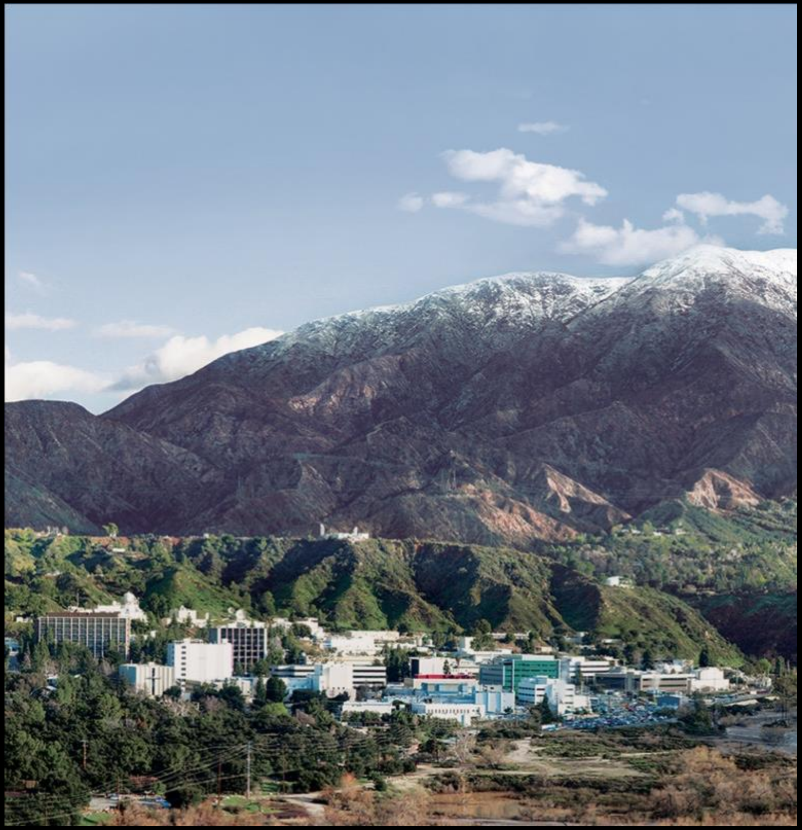
- Software Systems Engineer
  at NASA JPL
- Education:
  - Bachelor's of Science in Industrial and Systems
    Engineering (Georgia Tech)
- Staffed on two flight projects
  - Europa Clipper
  - Europa Lander
- MBSE Native: model-based engineering from start of career

Agenda

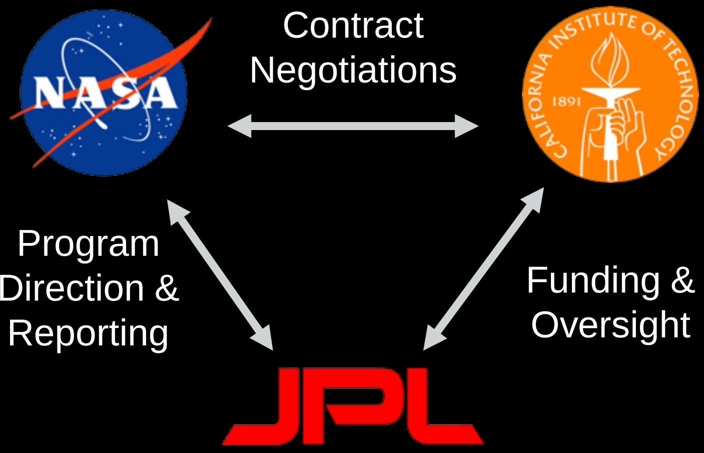
- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

NASA Jet Propulsion Laboratory (JPL)



- Located in Pasadena, CA
- NASA-owned *"Federally-Funded Research and Development Center"*
- University-operated
- ~5,000 employees



You May Know Some of Our Missions…

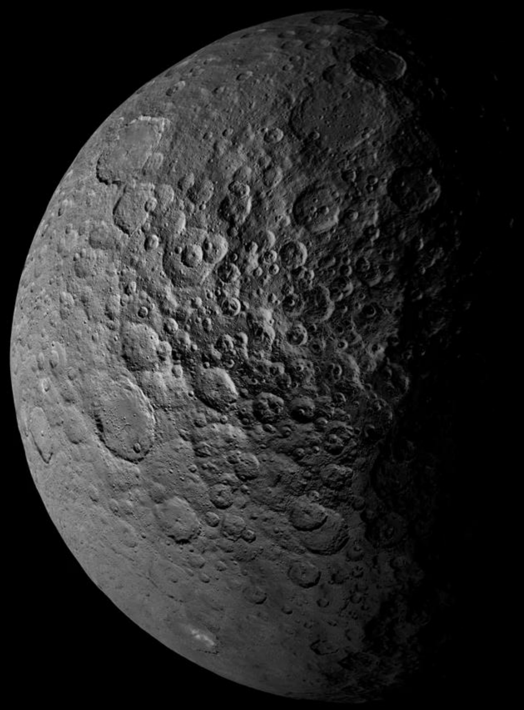
Voyager 1 & 2 (1977)

JPL’s Mission is Robotic Exploration

Mars Science Laboratory (2012)

Computer Aided Engineering (CAE)

- Computer Aided Engineering provides the Laboratory's Engineering Staff and Scientific communities with tools and technical expertise
- Four Environments:
  - Systems Environment
  - Software Environment
  - Mechanical Environment
  - Electrical Environment



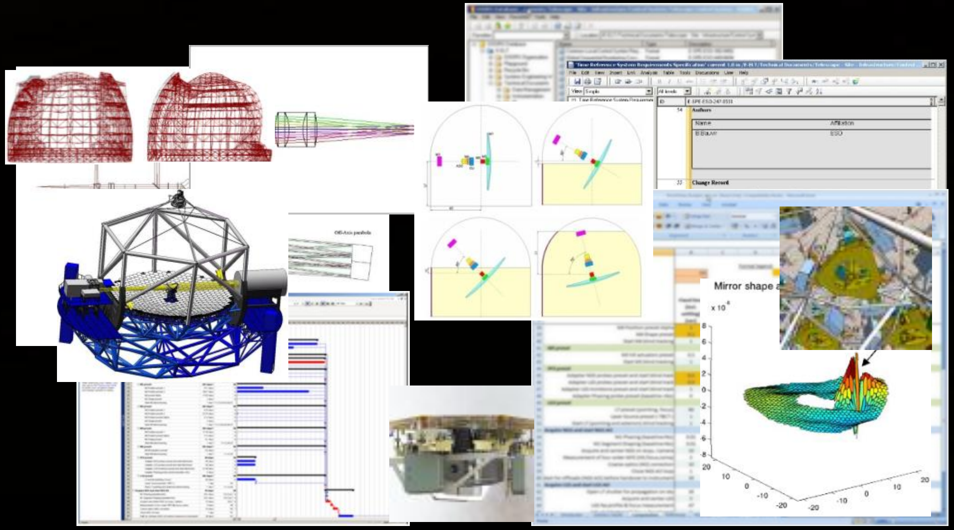
Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

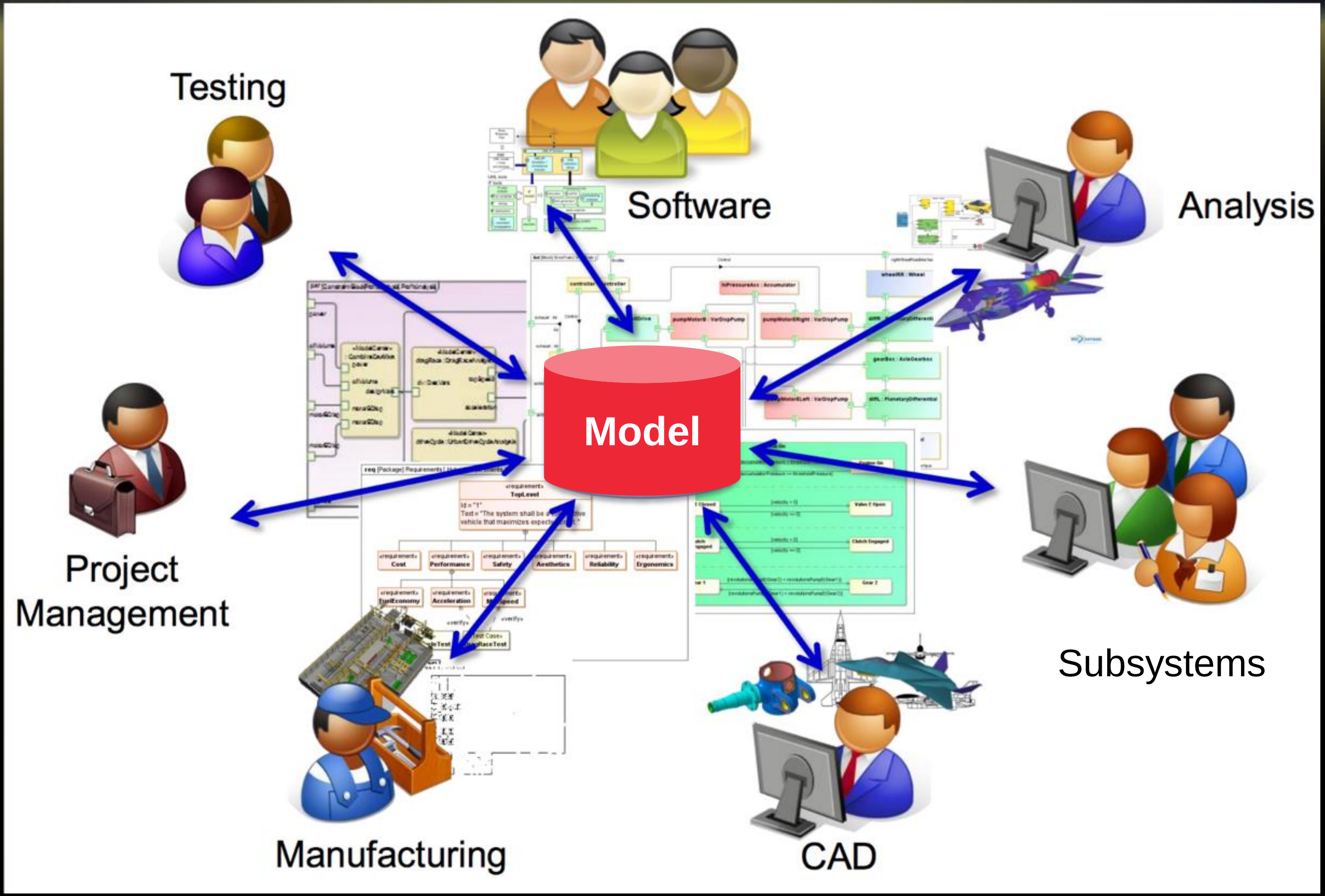
CAE Problem Statement

- Current State of Practice
  - Dispersed domain specific modeling (CAD, FEA, MATLAB)
  - Document-based artifacts related to models, but not connected
- Need for a Model-Based Engineering Environment
  - Tie system level models into existing models and modeling tools
  - Provide methods and tooling environment to support the effort



Systems Environment: Model-Based Approach

Information Management Across All Disciplines and the Life Cycle



Systems Engineering: Executable Approach

- Next phase of modeling emphasizes executable models to enhance understanding, precision, and verification of requirements
- Executable Systems Engineering Method (**ESEM**) augments the OOSEM activities by enabling executable models
  - ESEM defines executable SysML models that verify requirements
  - Includes a set of analysis patterns that are specified with various SysML structural, behavioral and parametric diagrams
  - Also enables integration of supplier/customer models and analysis

Applications of Model-Based Engineering at JPL

JPL is applying MBE practice in several projects

- Missions to Europa
  - Europa Clipper
  - Europa Lander
- Missions to Mars
  - Mars 2020
  - InSight
  - Mars Sample Return (MSR)
- Thirty Meter Telescope
- Ground Data Systems
- Psyche
- MAIA

Engineering Products

- MELs, PELs
- Resource allocation analysis
- System decomposition,
- Libraries / reusable models

Not just spacecraft missions! Not just early phases of design!

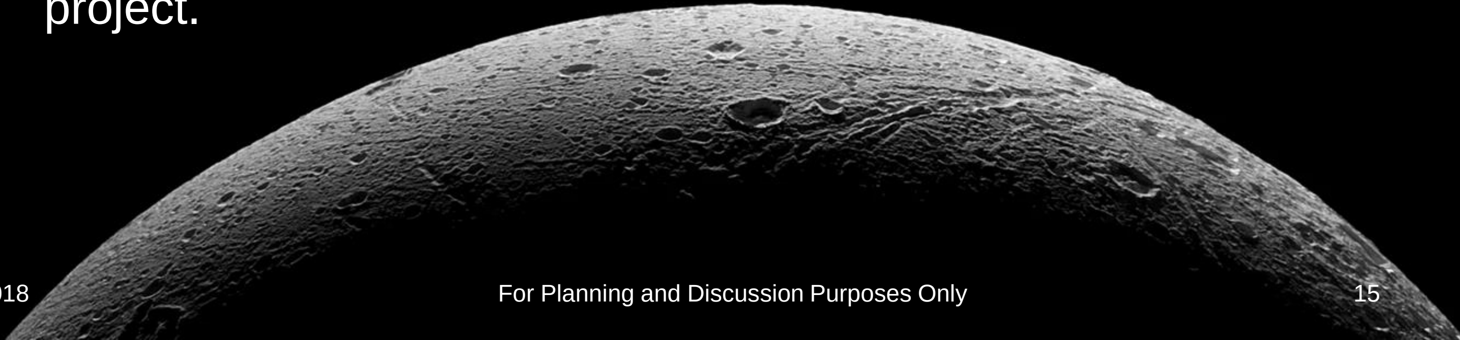
Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

OpenCAE Vision

- Provide an open portfolio in a shared environment that seamlessly connects engineers developing missions and systems.
  – Open - The portfolio that CAE provides is open in every sense of the sprit of open source. Our processes, code, apps, services and artifacts are accessible by JPL users as well as vendors and partners.
  – Shared - CAE is more than a collection of licenses and tools, its a shared environment for engineering. The diverse community of users, developers partners and vendors are able to contribute innovation and work more effectively by reducing the overhead.
  – Connected - the CAE Environment connects engineers allowing them to collaboratively construct and analyze the precision products needed to develop Missions and Systems at JPL using the CAE environment. This is done without the overhead of traditional manual exchanges of information. Engineers can connect with each other and find relevant engineering data and information reducing redundancy and increasing value of the engineering products and analysis produced by the flight project.

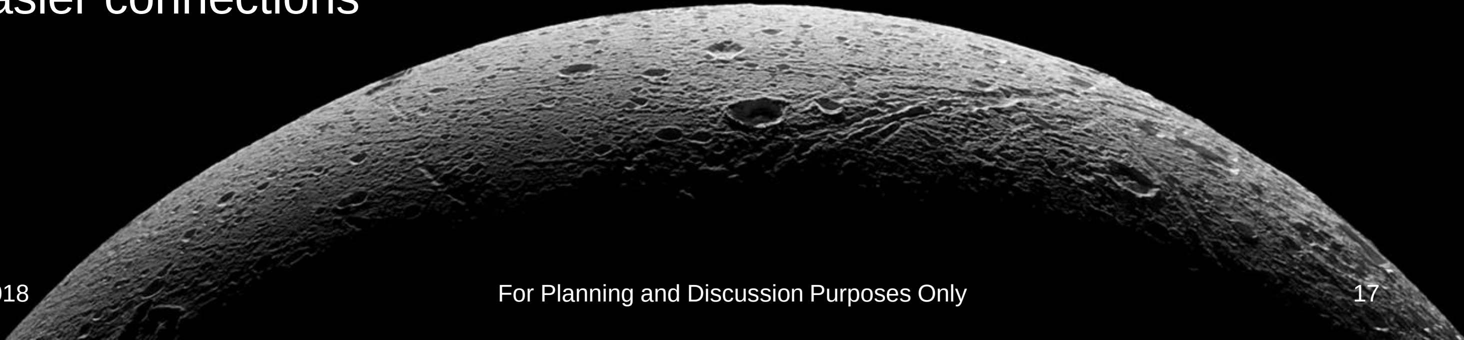


OpenCAE Mission

- Develop the CAE environment from a user centered architecture leveraging vendor partnerships using robust life cycle processes.
  - Vendor partnerships – CAE works closely with Vendors providing them crucial feedback and insight into how their products are serving the needs of engineers and developers
  - User centered architecture – to achieve the vision of Open CAE, the technical architecture for CAE is driven by the needs of the practitioners who use the environment and the needs of the projects that are served by it
  - Life-cycle process – the life-cycle processes for CAE provide the integrity of the the applications services and support provided by CAE

OpenCAE System: Overview

- Collection of engineering environments based on a Technology Portfolio is referred to as OpenCAE
- Provide a platform for these tools to work together in order to support JPL's various projects
- The integration platform provides the core to JPL's mission engineering environment allowing to tracking relations between heterogeneous data sources in a linked data architecture
- Evolution of those engineering environments is controlled through case studies
- Incorporate tooling from systems, software, mechanical, and electrical domains
- Lifecycle support for these tools
- Includes configuration management, archiving, business process implementation, and review support
- Emphasize standards for data interchange such as REST to provide for easier connections



OpenCAE System: Embedded Roles

- CAE provides the same environment to all its customers (engineers and scientists)
- Embedded roles work directly on projects to adapt the standard environment specific to the project goals or methodology
- Embedded roles capture needs in general case studies which inform the CAE architecture

Europa Clipper Embedded Role

- Need:
  - Formalize analysis workflows related to the Clipper Flight System
  - Want to capture the workflows in a model, but also want them to be executable
- Approach:
  - Use Phoenix MBSEPak plugin for MagicDraw to translate the workflow parameters into Phoenix ModelCenter
  - Configure ModelCenter to use shared components in the Analysis Library of ModelCenter Cloud

Europa Clipper Embedded Role

- Need:
  - Workflows need to publish artifacts to CAE services (MMS, TES, Artifactory)
- Approach:
  - Express the REST API endpoints of these servers in OpenAPI standard specification
  - Use Swagger codegen to generate clients for specific analysis environments
    - Mathematica, MATLAB, Python, Java
    - More than 20 other languages available

Europa Lander Embedded Role

- Need:
  - Generate orderly and palatable diagrams from a system model describing the Lander
  - SE products should never be out of sync with the system model
- Approach:
  - Leverage Tom Sawyer plugin for MagicDraw development effort
  - Supply requirements directly from the project to the vendor
  - Coordinate with CAE development team on the use case for Tom Sawyer integration with DocGen and View Editor

OpenCAE System: User Communication

- Mailing lists generated by tool license use
- Slack channels per each tool for general questions (with vendors)
- Technical Working Groups held biweekly with vendors for tool-specific questions
- OpenCAE Systems Environment Team Office Hours held biweekly for general questions and support

OpenCAE System: User Centered Design

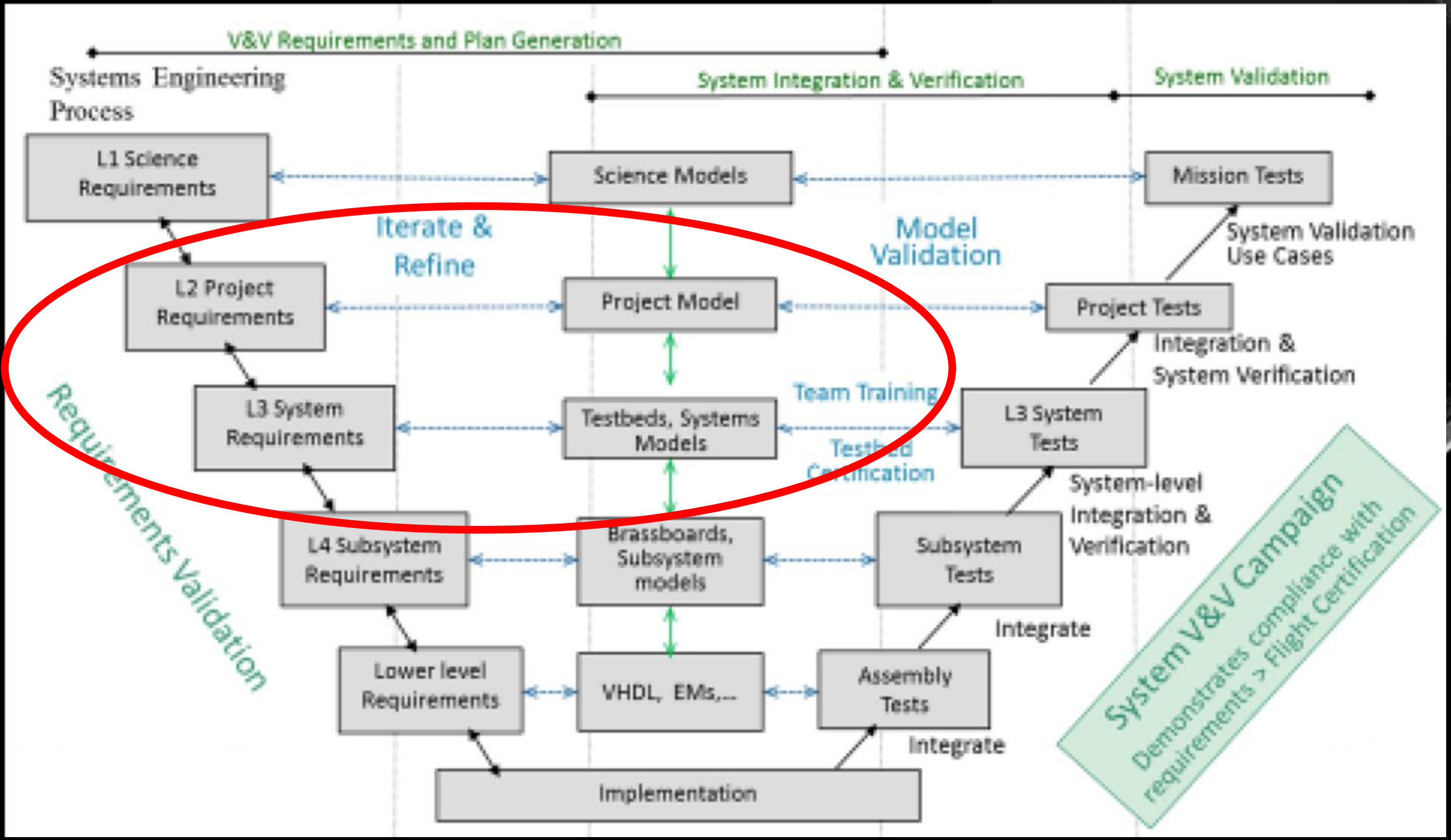
- User Centered Design steers the development of the OpenCAE infrastructure
- Continuous communication with users to understand their experience in the OpenCAE environments
- Users evaluate solutions before they are implemented
- Following standard UX practices

CAE Systems Environment

Support SE Activities:

- Requirements Management
- Interface Management
- Design Management
- Trade Studies
- Interdisciplinary Integration
- Analysis Management
- Resource Management

Scope of the CAE Systems Environment



CAE Systems Environment

Case Studies	Realization
Systems Design Management	MagicDraw, View Editor, MapleMBSE
Systems Resource Management	Phoenix ModelCenter, Cameo Simulation Toolkit, Systems Tool Kit
Interdisciplinary Integration	Syndeia, Cameo Datahub
Viewing and Reporting	Tom Sawyer, View Editor
Systems Analysis Management	Phoenix ModelCenter, Platform for Modeling Analysis (PMA)

OpenCAE Environments and Technology Portfolio

CAE Disciplines

Collections of tools and resources for engineering disciplines

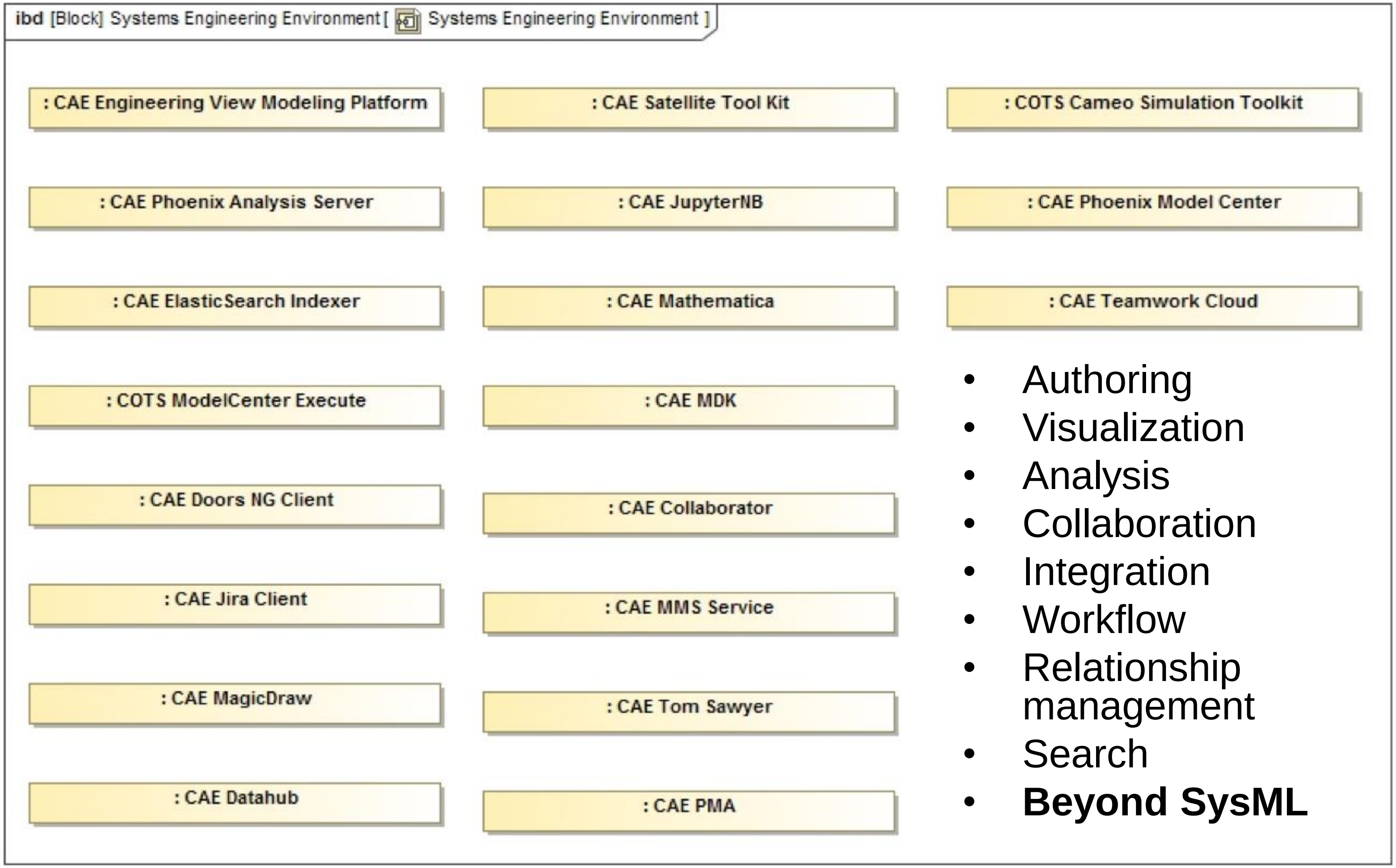
Electrical

Mechanical

Software

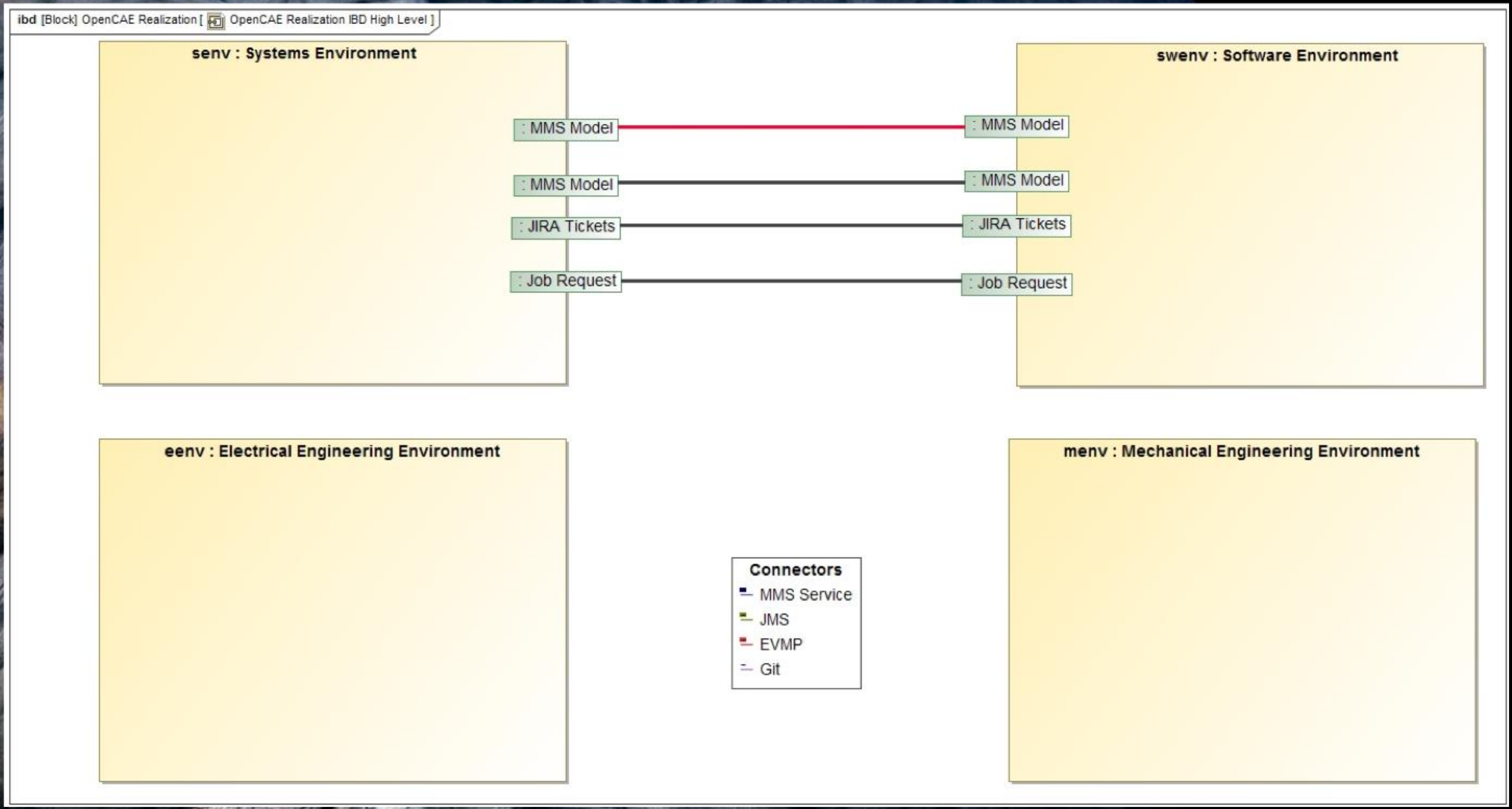
Systems

Systems Environment Tools

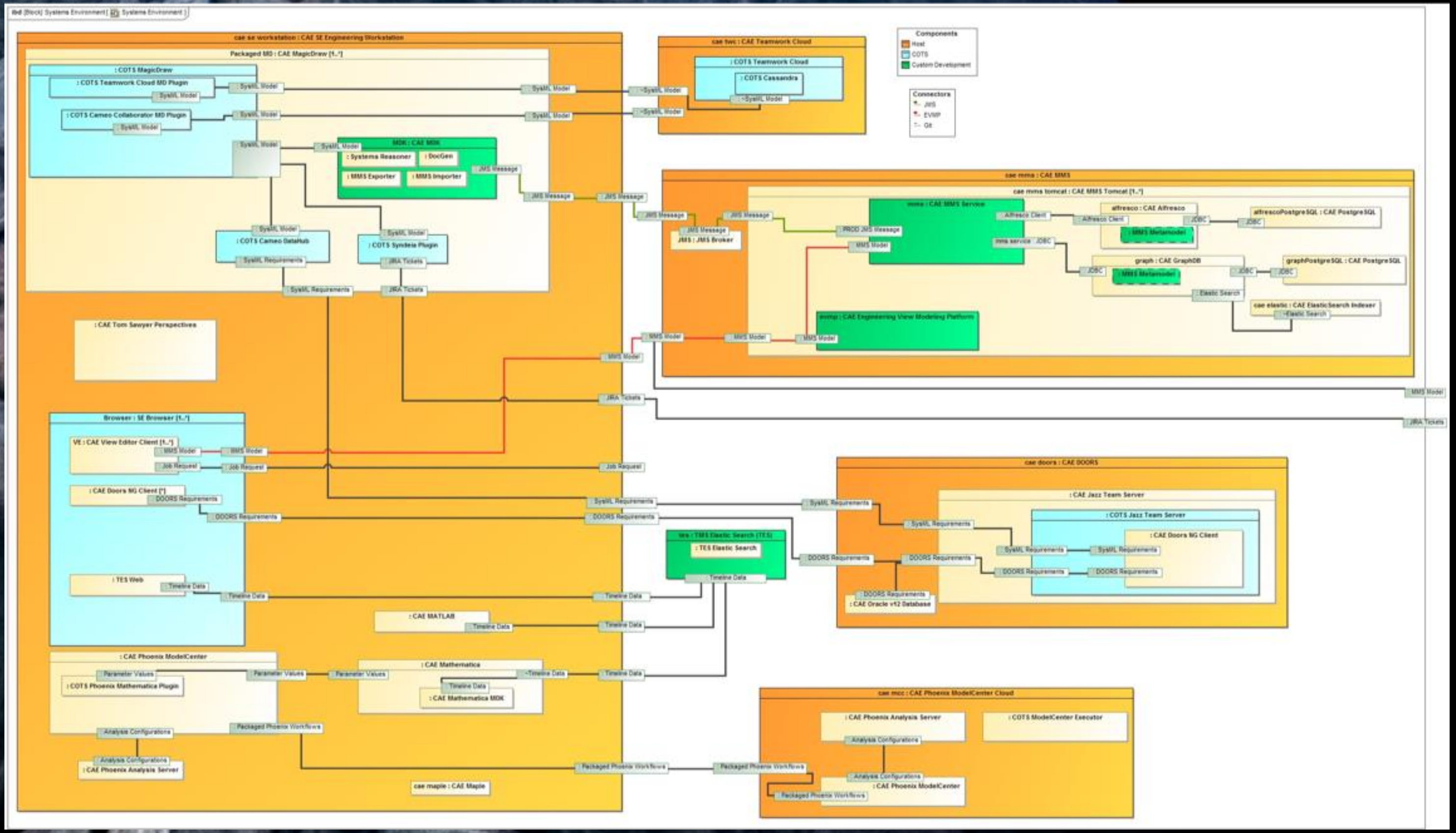


- Authoring
- Visualization
- Analysis
- Collaboration
- Integration
- Workflow
- Relationship management
- Search
- **Beyond SysML**

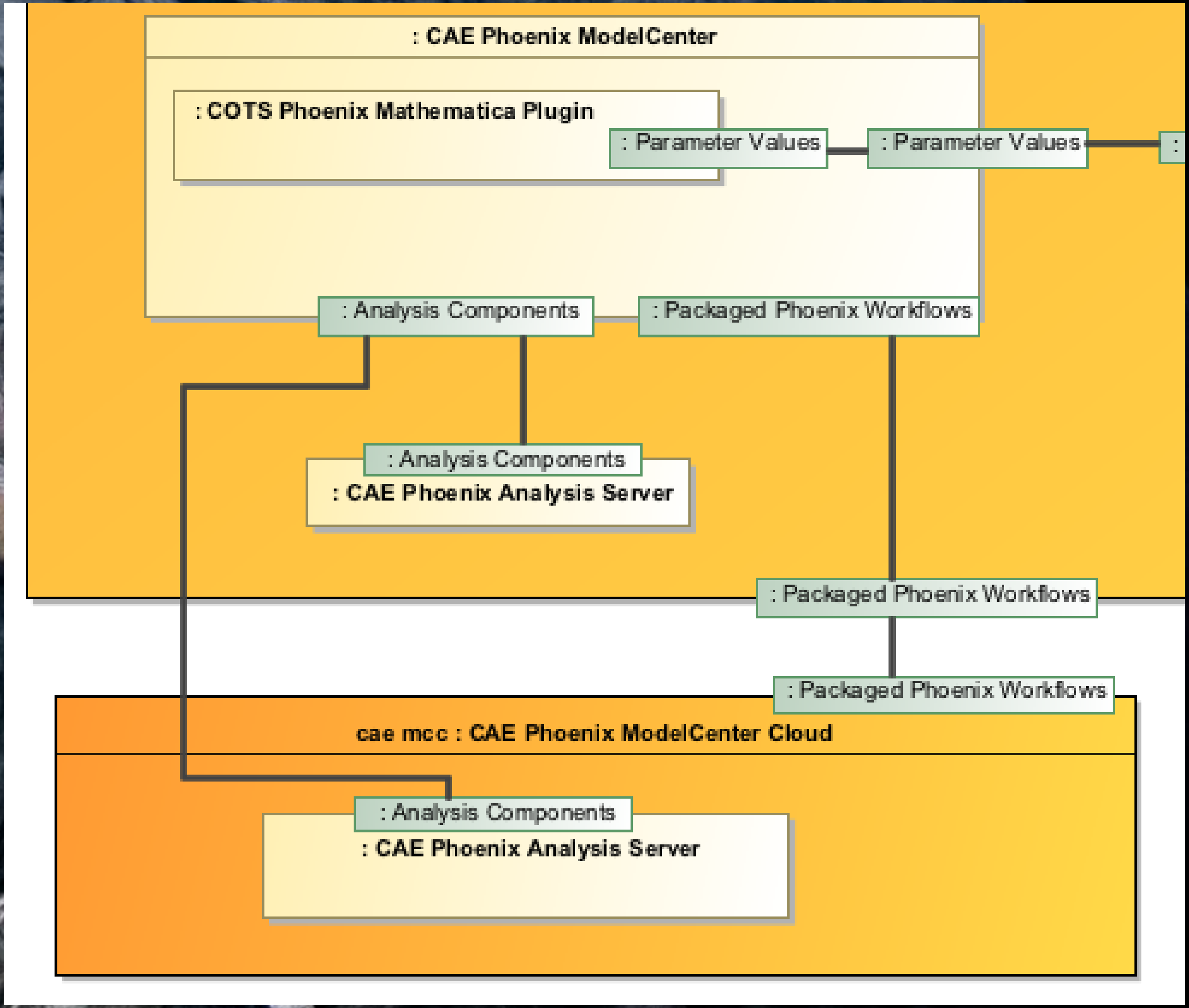
Interactions Between CAE Environments



Interactions Within CAE Systems Environment



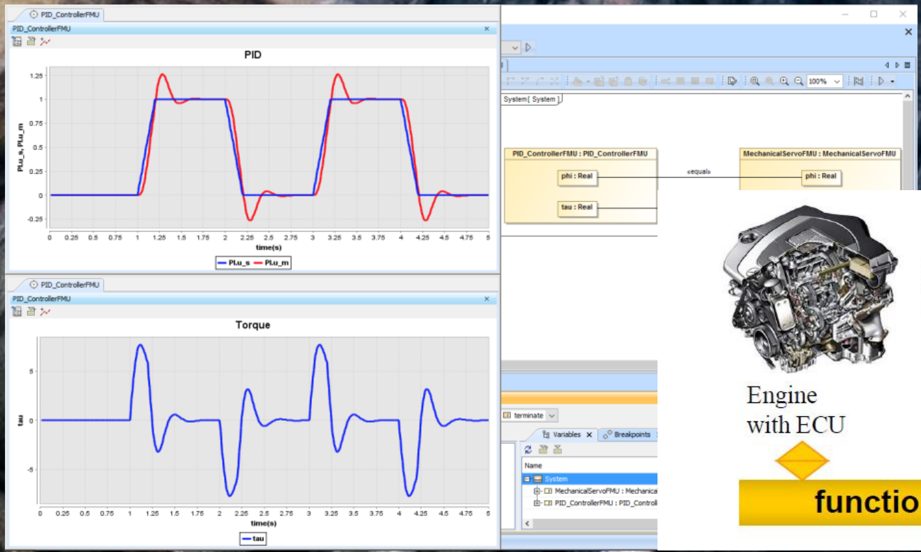
Vendor Spotlight: Phoenix Integration

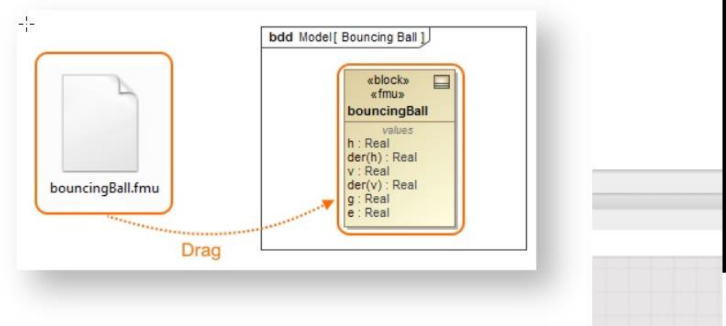


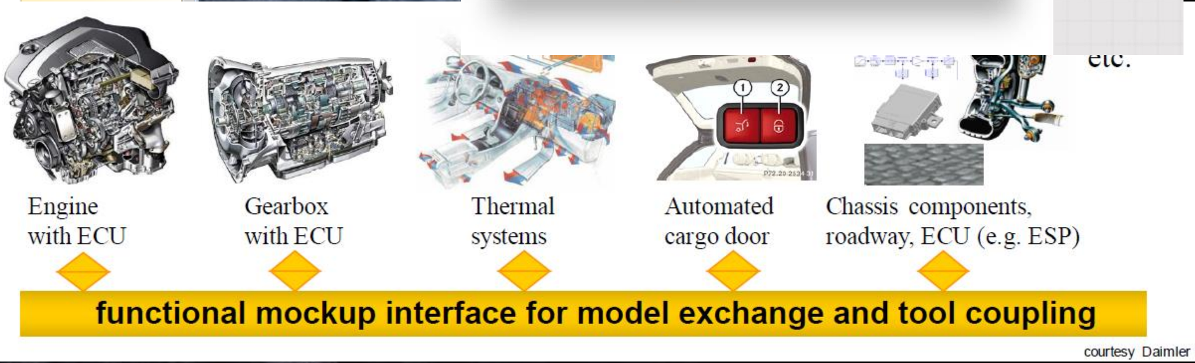
Standardized Co-Simulation

- The Functional Mock-up Interface (or FMI) defines a standardized interface to be used in computer simulations to develop complex cyber-physical systems
- Integration with System Level behavior model

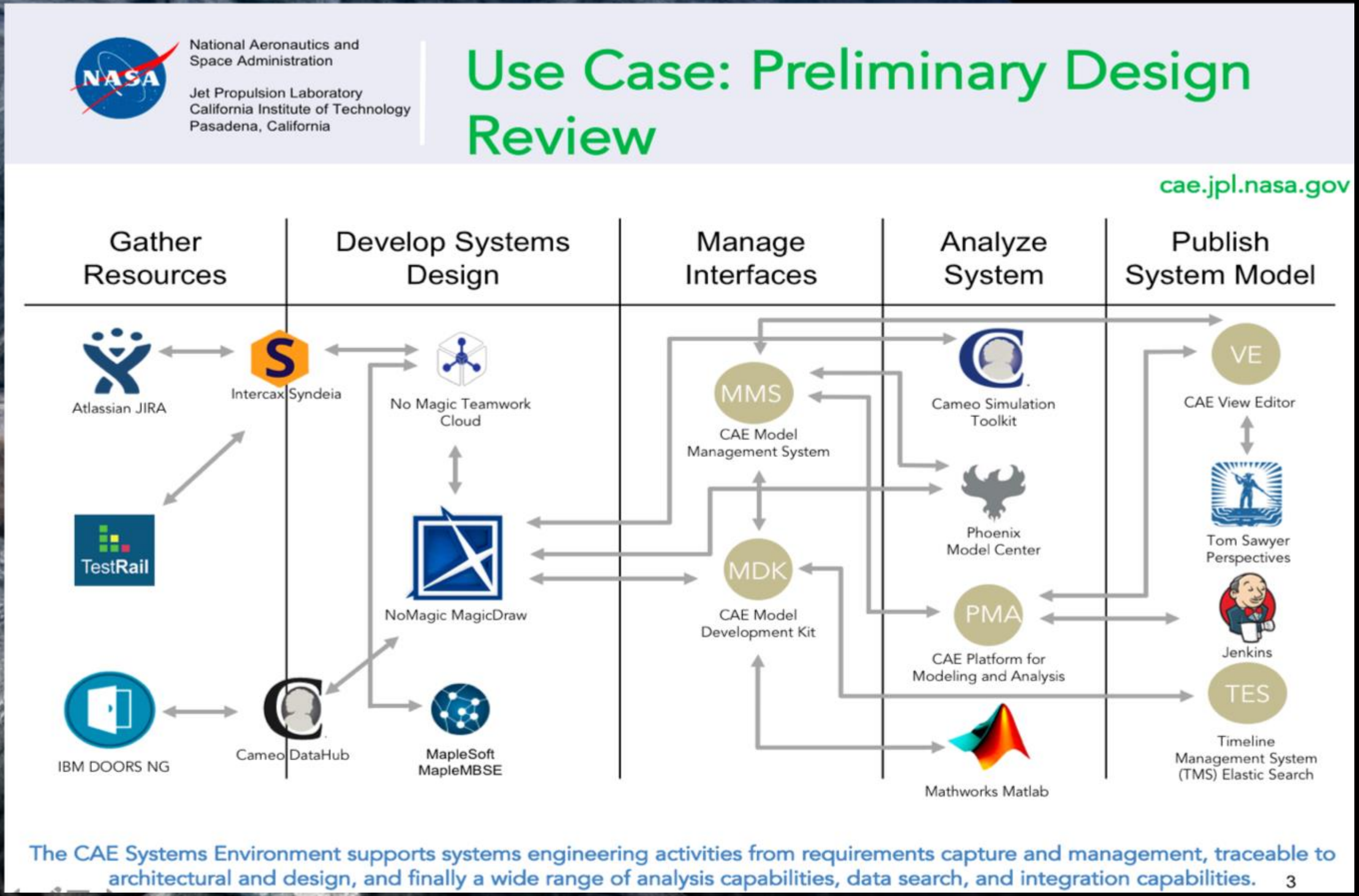
Tools supporting FMI	FMI Version	ModelExchange		CoSimulation		Notes
		Export	Import	Slave	Master	
Cameo Simulation Toolkit	FMI_1.0		Available		Available	FMUs can be imported, represented, connected and co-simulated in SysML models.







Systems Environment Use Case



Systems Environment Integrations: What Has Worked

- Vendor solutions
  - Connections between vendor servers
  - Managed Services
- Server-side operations preferred
  - Easier to update a server than to push clients
- Speak the same language (SysML, FMI)
- Swagger REST API
  - Generate Swagger clients for users’ preferred languages
  - Enforces OpenAPI on environment services

Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

openmbee.org

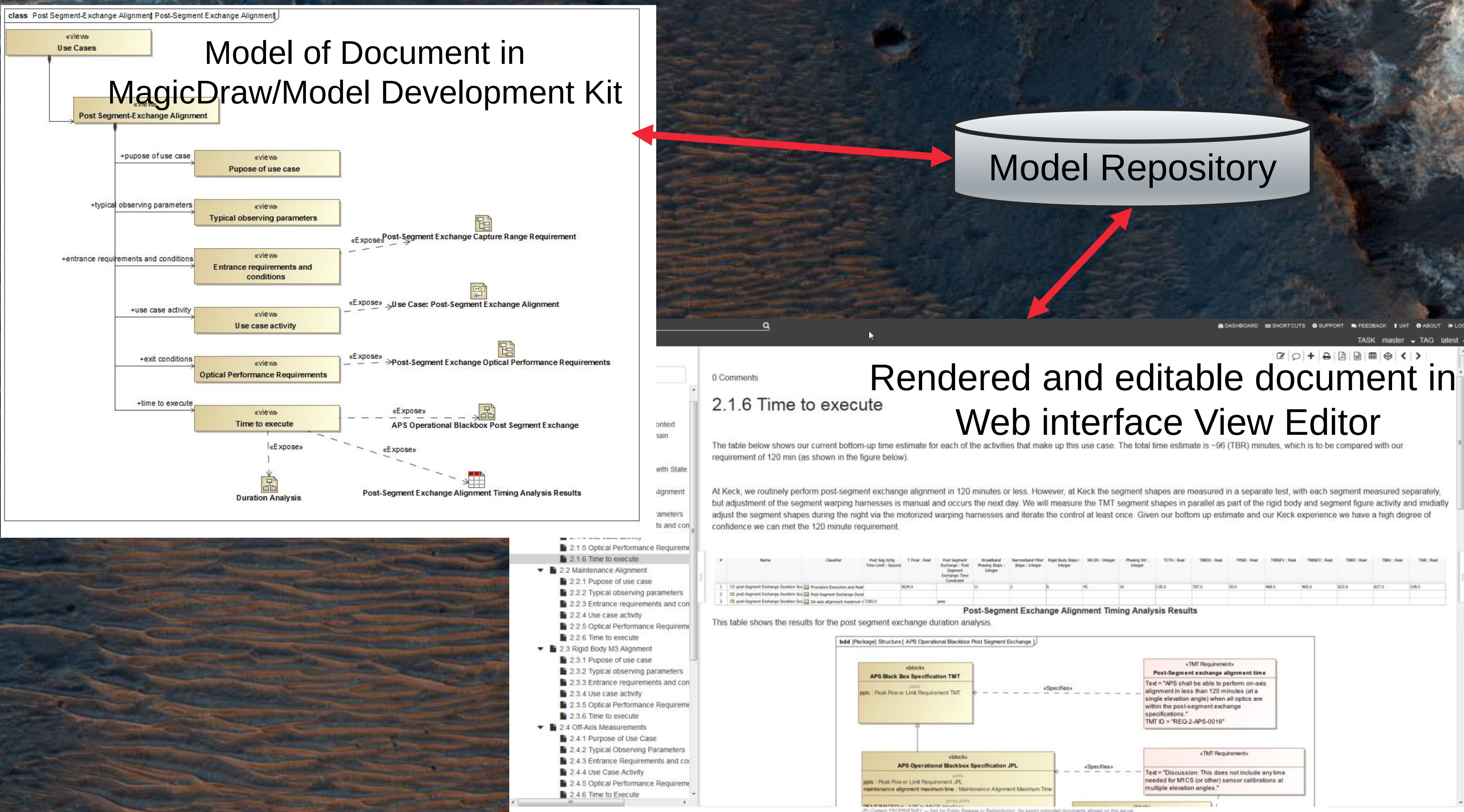
Open Model Based Engineering Environment

- OpenMBEE is a community for open-source modeling software and models
  - Number of open source software activities
  - Number of open source models
- JPL is a participant and adopter of OpenMBEE software and models

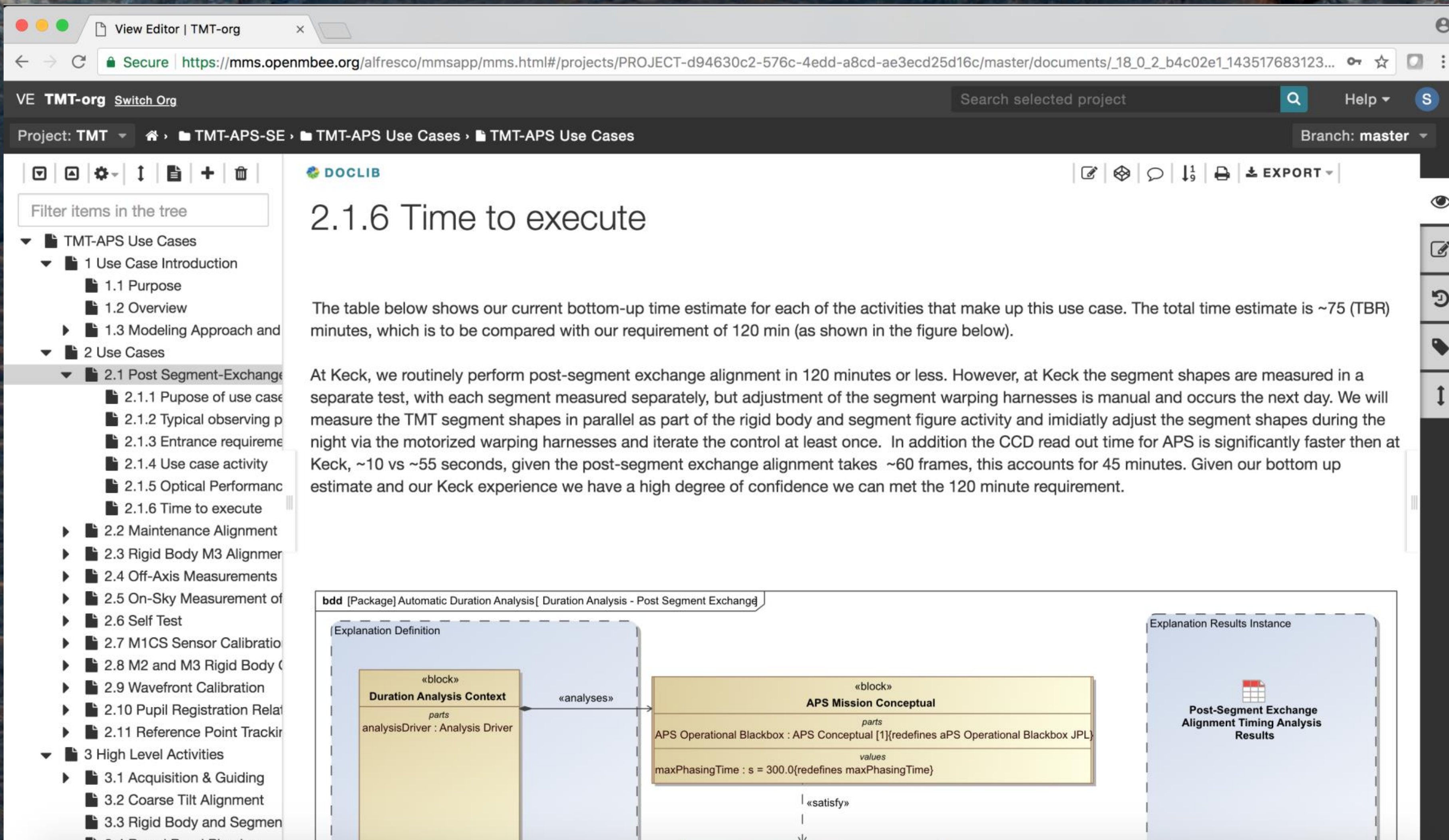
MMS, MDK, and View Editor

- The MMS model repository supports the following features:
  - Basic Infrastructure for Version, Workflow, Access Control
  - Flexibility of model content
  - Support for Web Applications and Web-based API access
  - Integration across engineering and management disciplines
- MMS is accessible from:
  - Rich SysML desktop clients like MagicDraw (via MDK)
  - Light-weight web-based clients like View Editor
  - Mathematical computation programs like Mathematica
  - Any tool that can utilize RESTful web services
- View Editor enables users to interact with SysML models within a web-based environment
  - System models are constructed, queried and rendered following the view and viewpoint paradigm
- View Editor implements the MMS REST API to provide a web environment to create, read, and update model elements

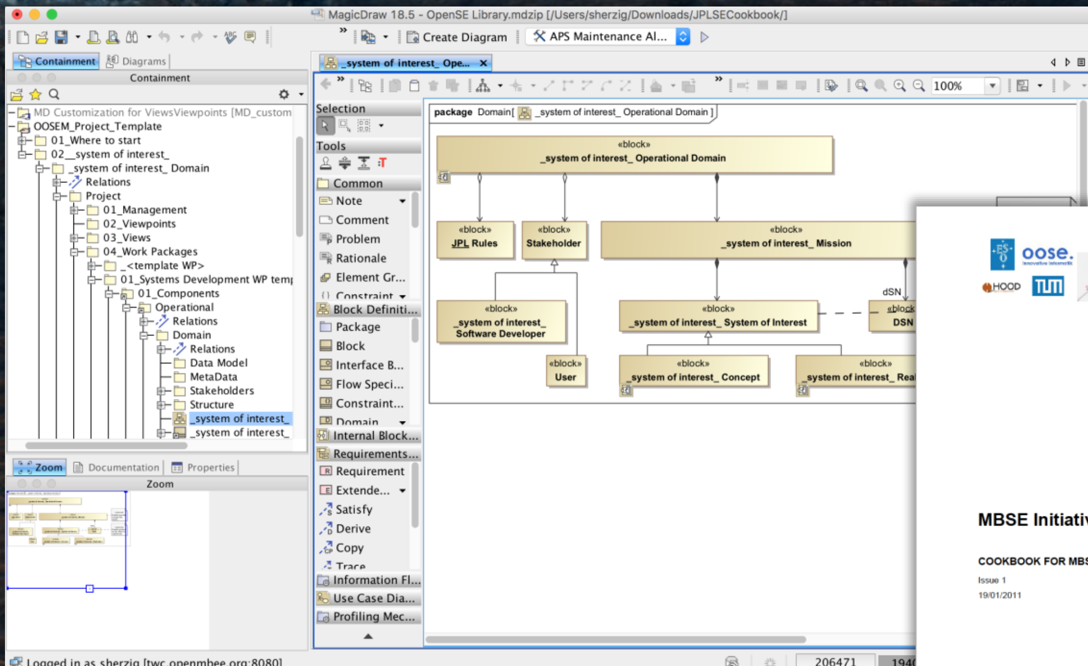
Core Integration of MMS, MDK, and VE



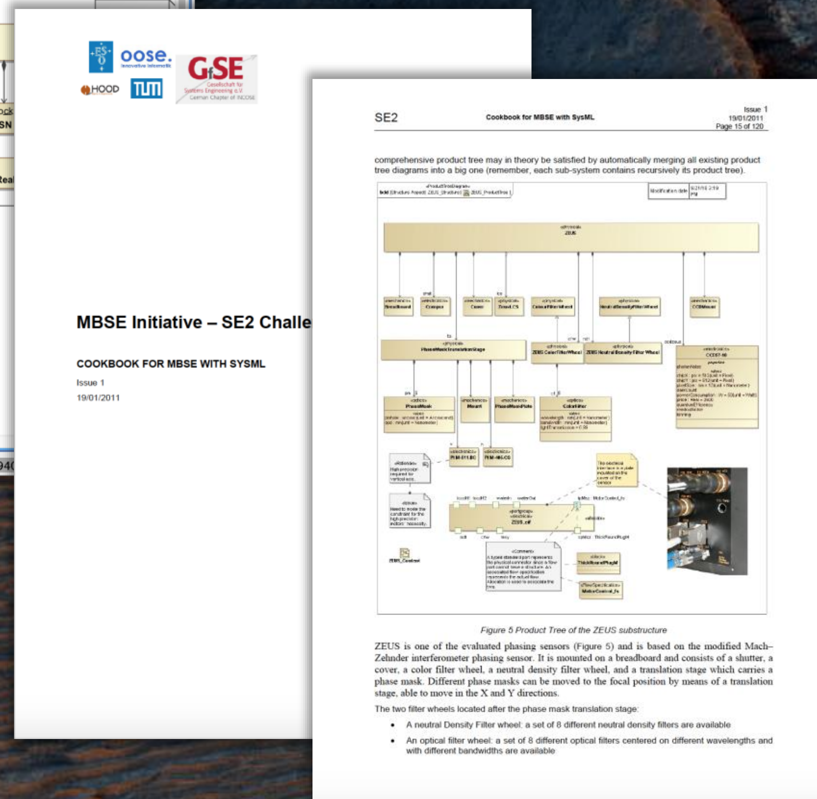
Document Generation Results on View Editor



OpenSE Cookbook and Template Model

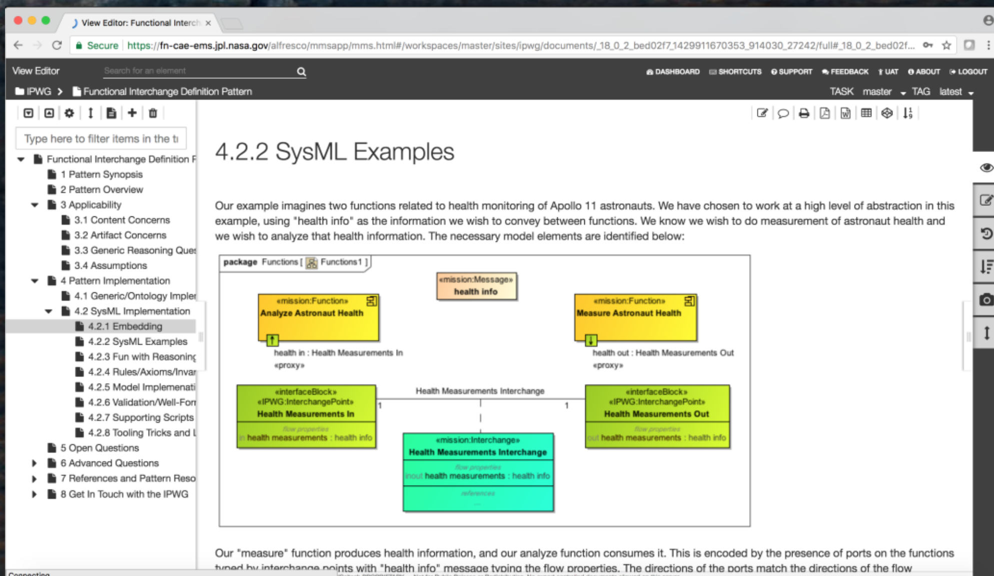


“Cookbook” for modeling methodology & patterns



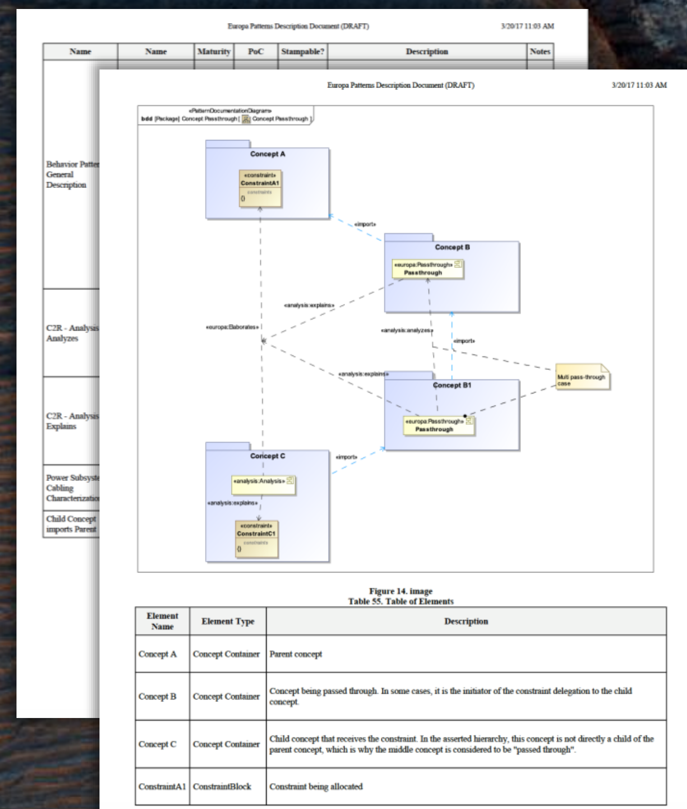
Template models to be used by projects as a starting point, with recommended organization, model libraries, etc.

SysML Modeling Patterns Development



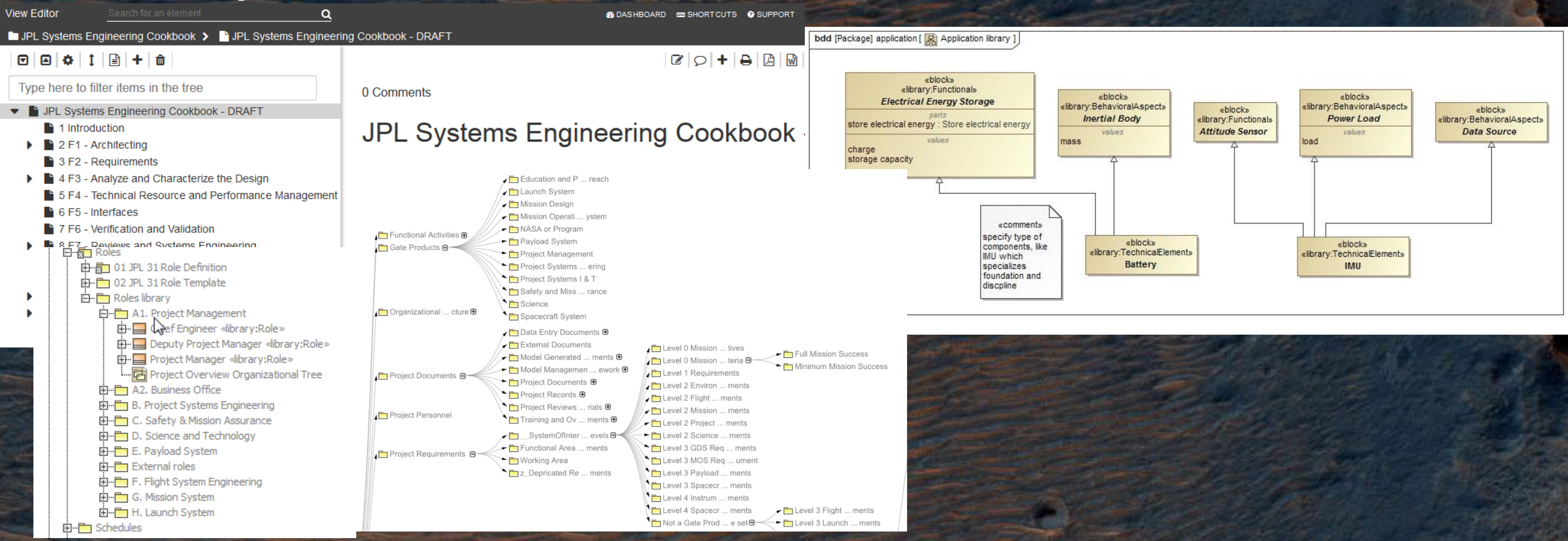
Project-independent modeling patterns as guidelines from overarching line organization

Project-specific modeling patterns for common modeling tasks

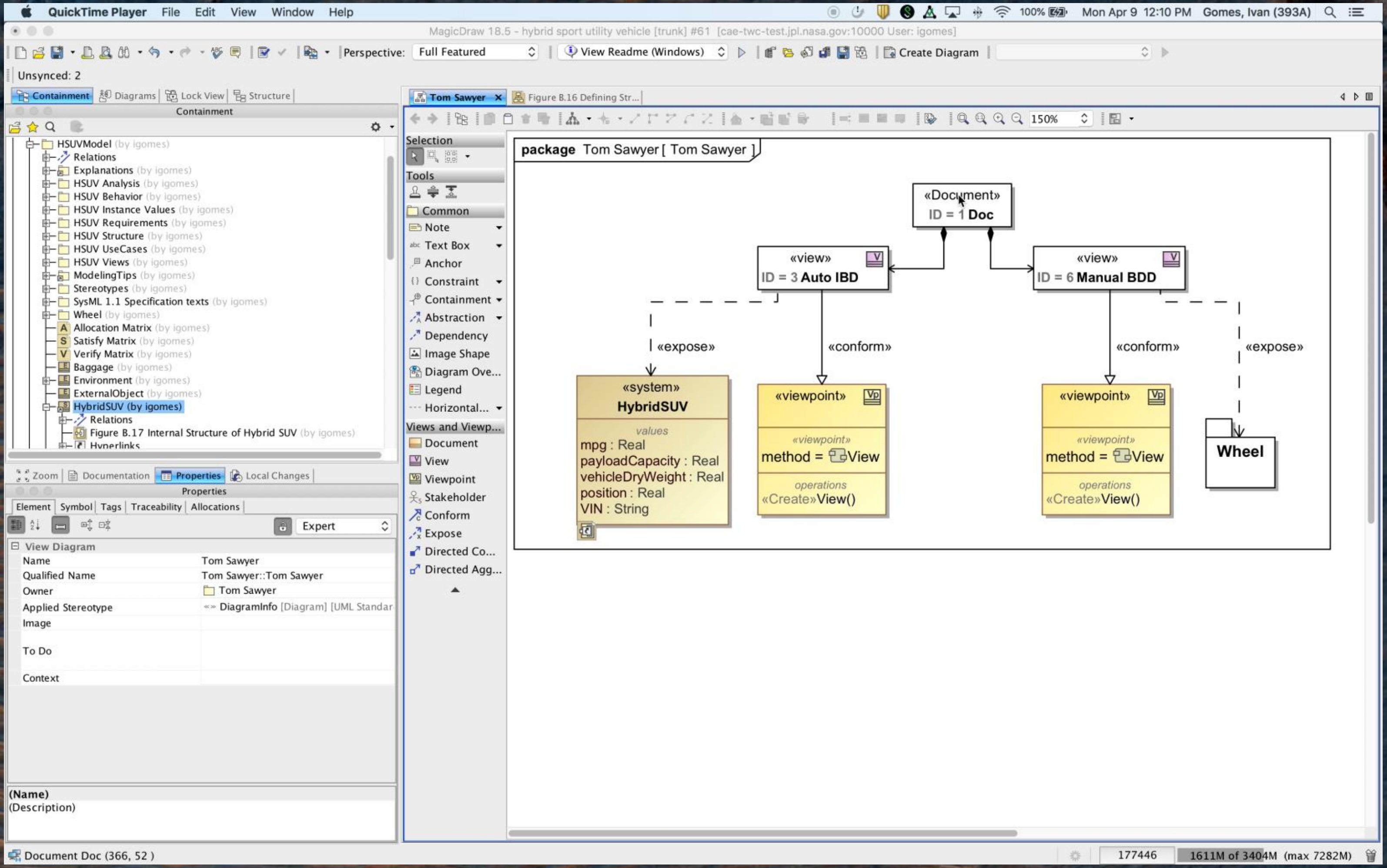


JPL SE Cookbook

- Collection of processes, practices, patterns to support Systems Engineering with model based techniques specific to JPL
- Organized according to 10 JPL SE functions
- Provides a set of SysML libraries, e.g. WBS Elements, Project Roles, Functional Elements, Model structure



DocGen – Tom Sawyer Integration for Query-Based Visualization

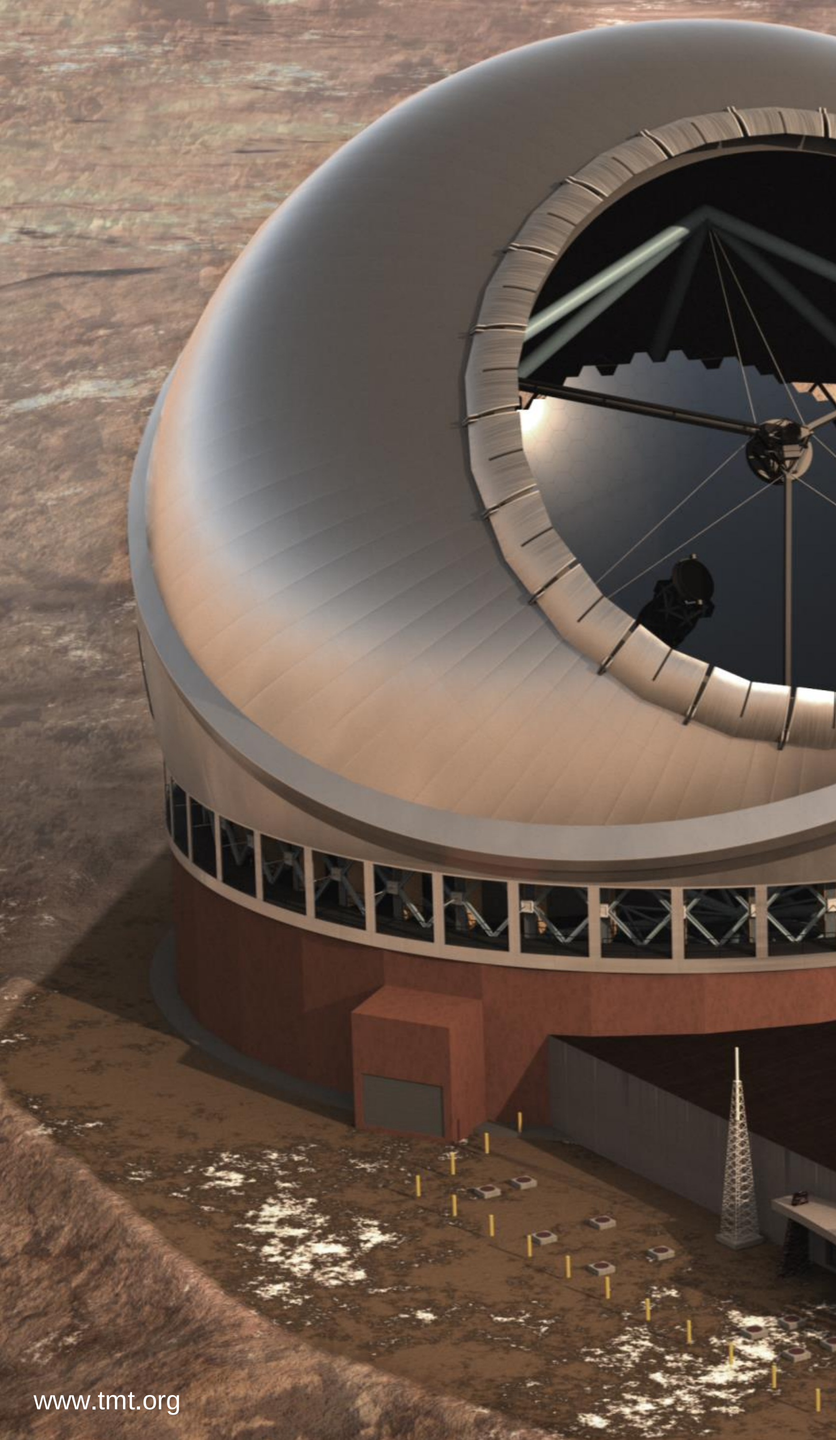


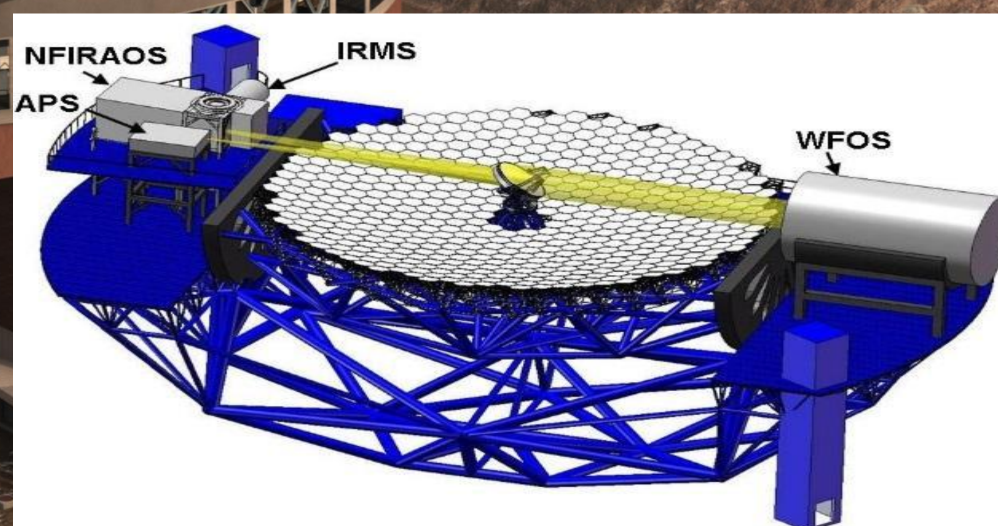
Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

The Thirty Meter Telescope (TMT) Project

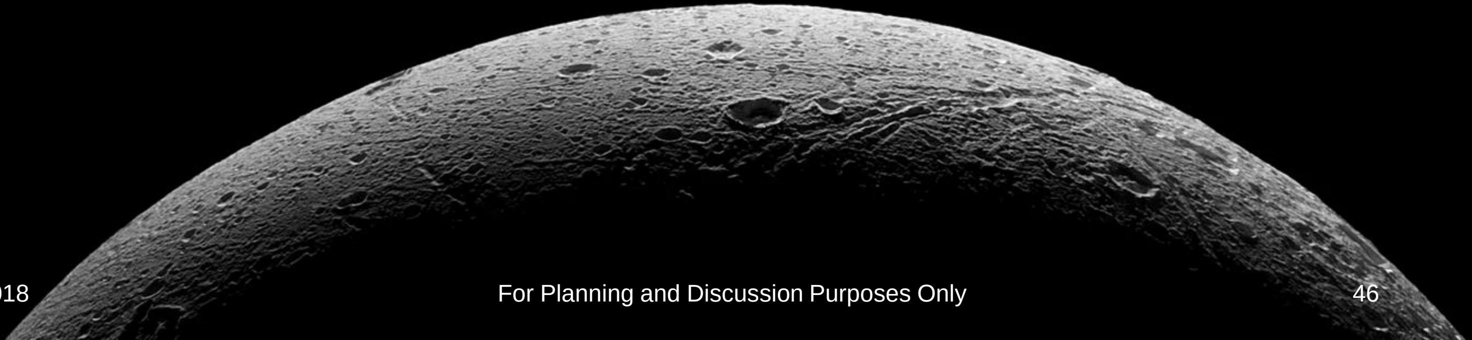




www.tmt.org

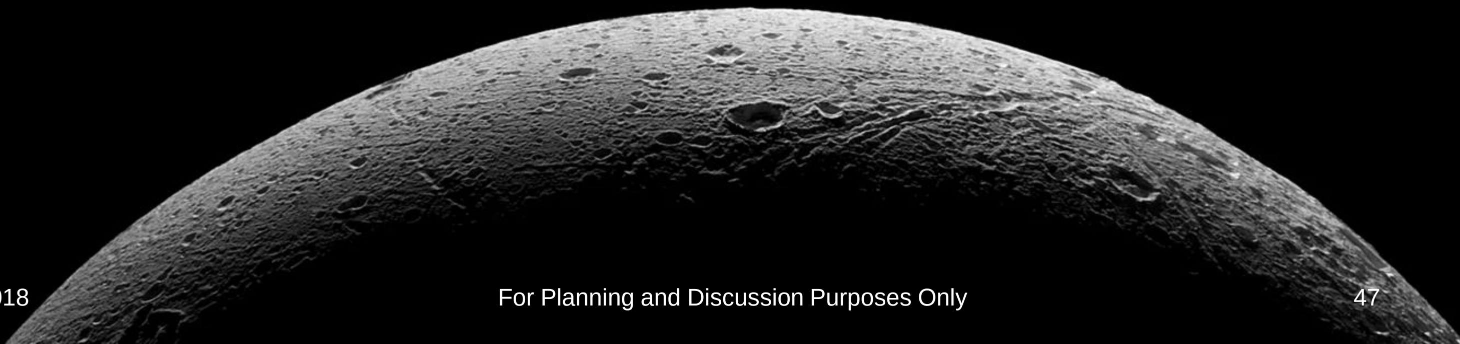
TMT MBSE Objectives

- Define an **executable SysML model**
- Use the model to **analyze the system design and verify requirements** on power consumption, mass, duration, pointing errors, etc.
- Produce **engineering documents**
  - Requirement Flow Down Document
  - Operational Scenario Document
  - Design Description Document
  - Interface Control Documents
- Use **standard languages and techniques, and COTS tools where practical** to avoid custom software development

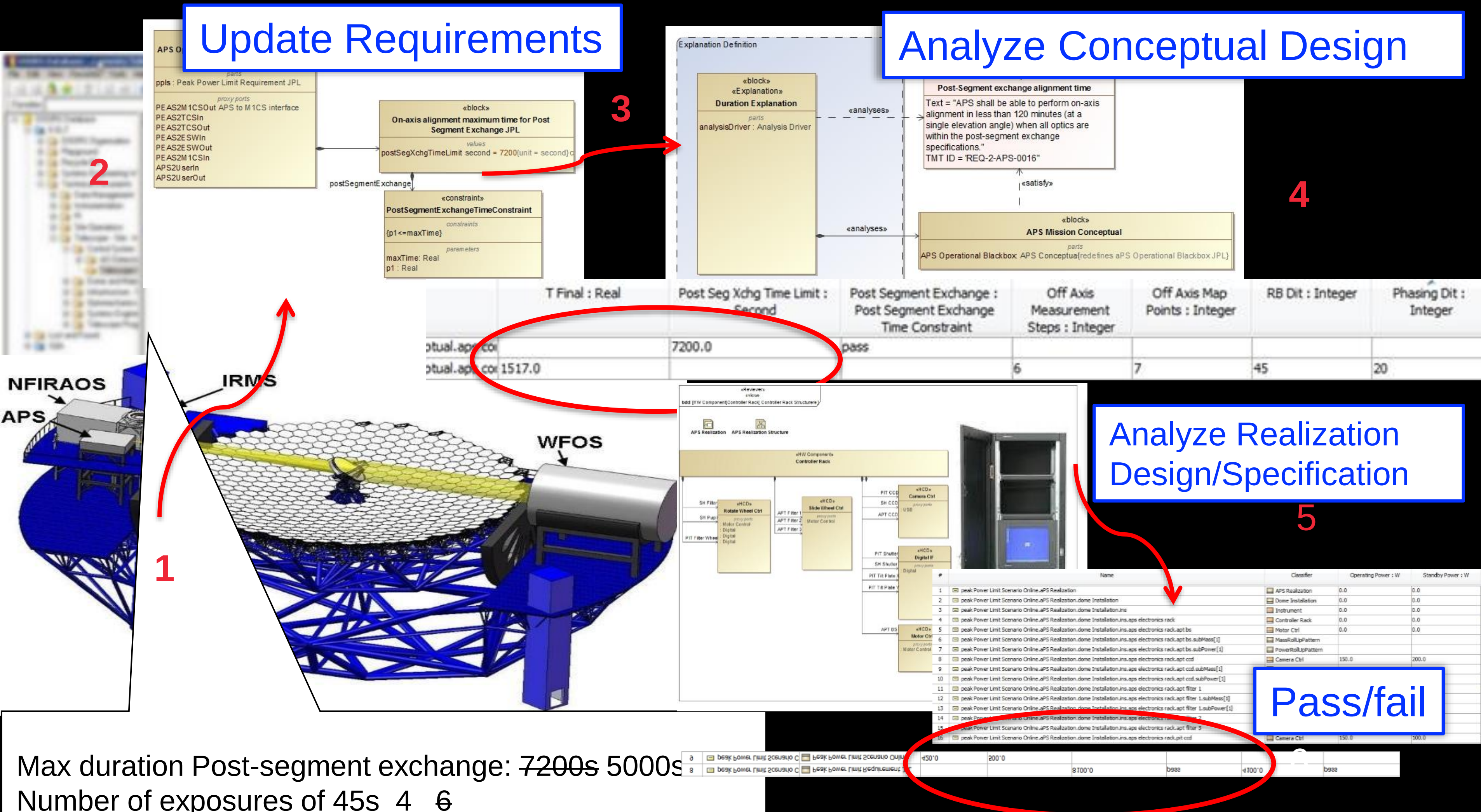


Modeling Approach

- **Object-Oriented Systems Engineering Methodology (OOSEM)**, but with additional activities focusing on building an executable model
- Use case driven model development
- Challenges:
  - JPL is a **supplier** for a number of subsystems of the TMT (the **customer**)
  - Model is used by a number of teams, including TMT



Analysis of Architecture and Design



OCD, Requirements, ICD, DDD

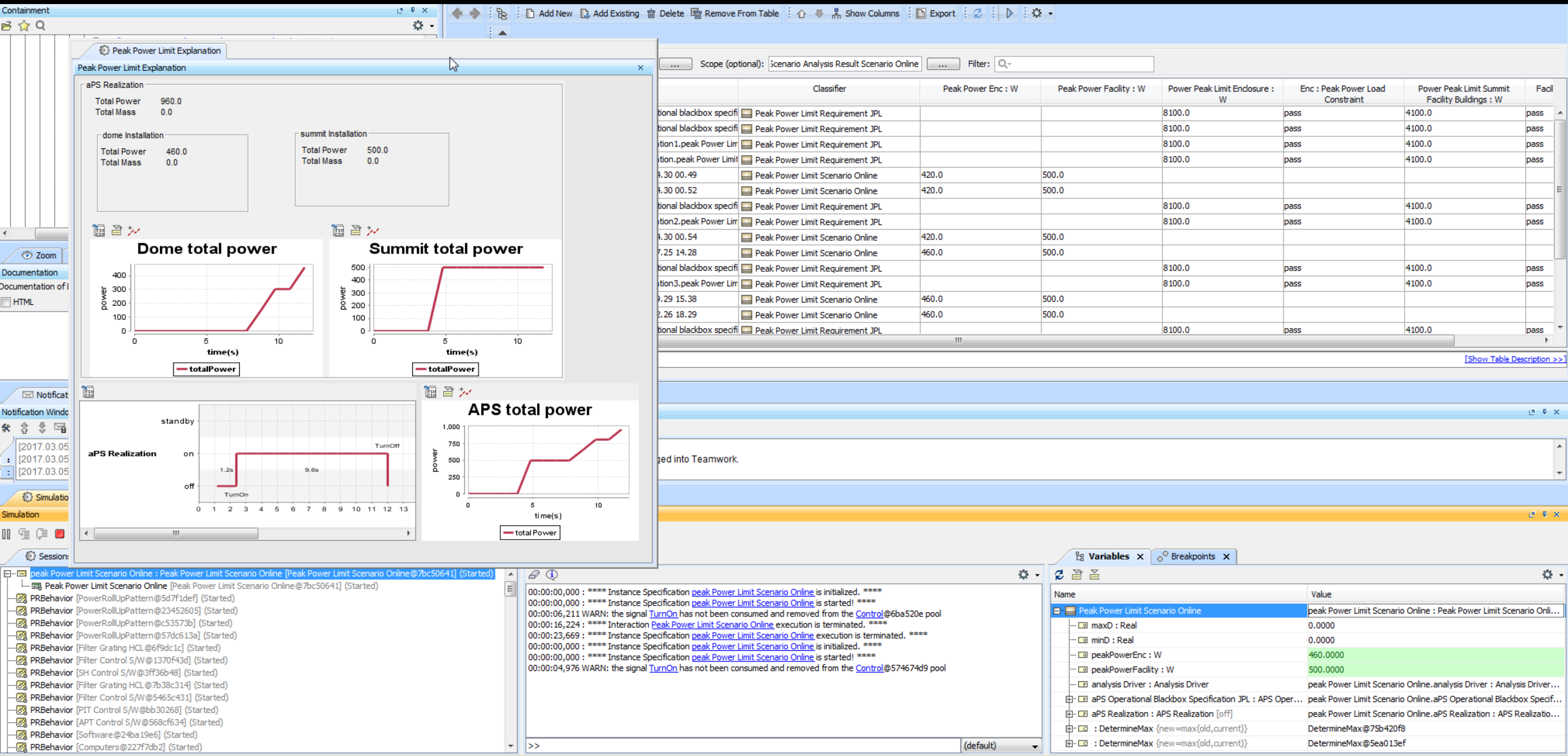
Max duration Post-segment exchange: ~~7200s~~ 5000s

Number of exposures of 45s 4 ~~6~~

Max peak power consumption in dome: 8.5kw ~~8.1kw~~

Number of motors with 50W 10 1~~2~~

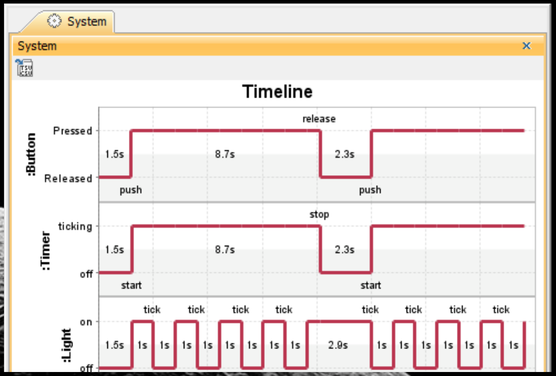
Power Analysis



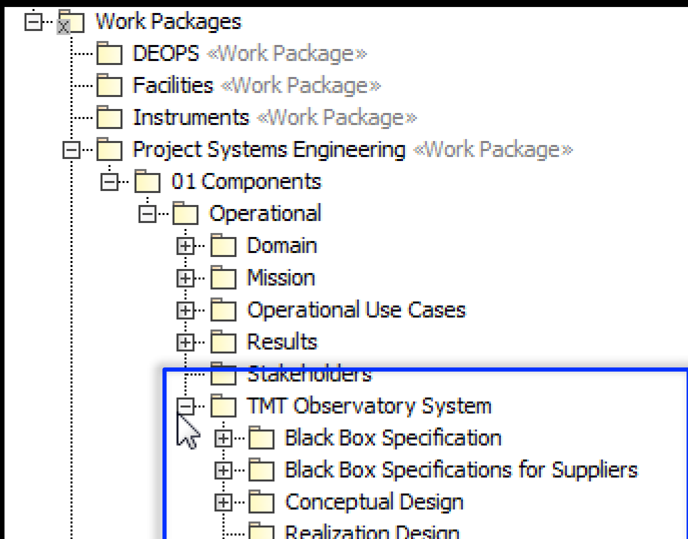
Run Analysis

- Run a configured analysis with a simulation engine on the initial conditions to get the final conditions
- Produce the following views on final conditions
  - **Table** showing final analysis values (e.g., peak power) and the constraint's pass/fail status for each scenario
  - **Timelines**: state changes for components over time
  - **Value profiles**: total rolled up values over time

#	Name	Classifier	T Final : Real	Ph
1	calibrations Duration S	Calibrations Duration S		
2	calibrations Duration S	APS Conceptual		
3	calibrations Duration S	Procedure Executive ar	8466.0	11

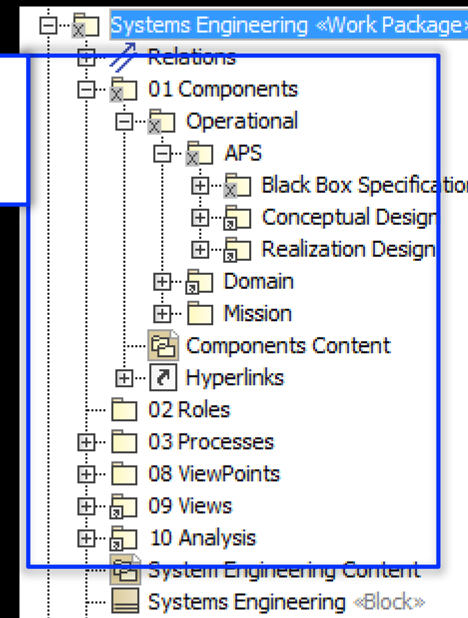


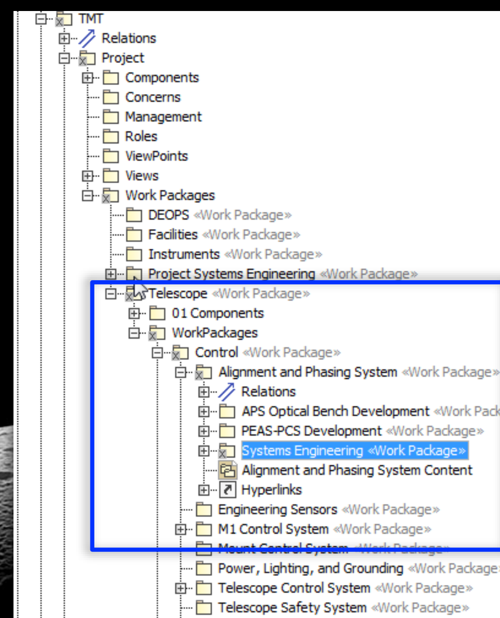
Package Organization



Customer / supplier relationship

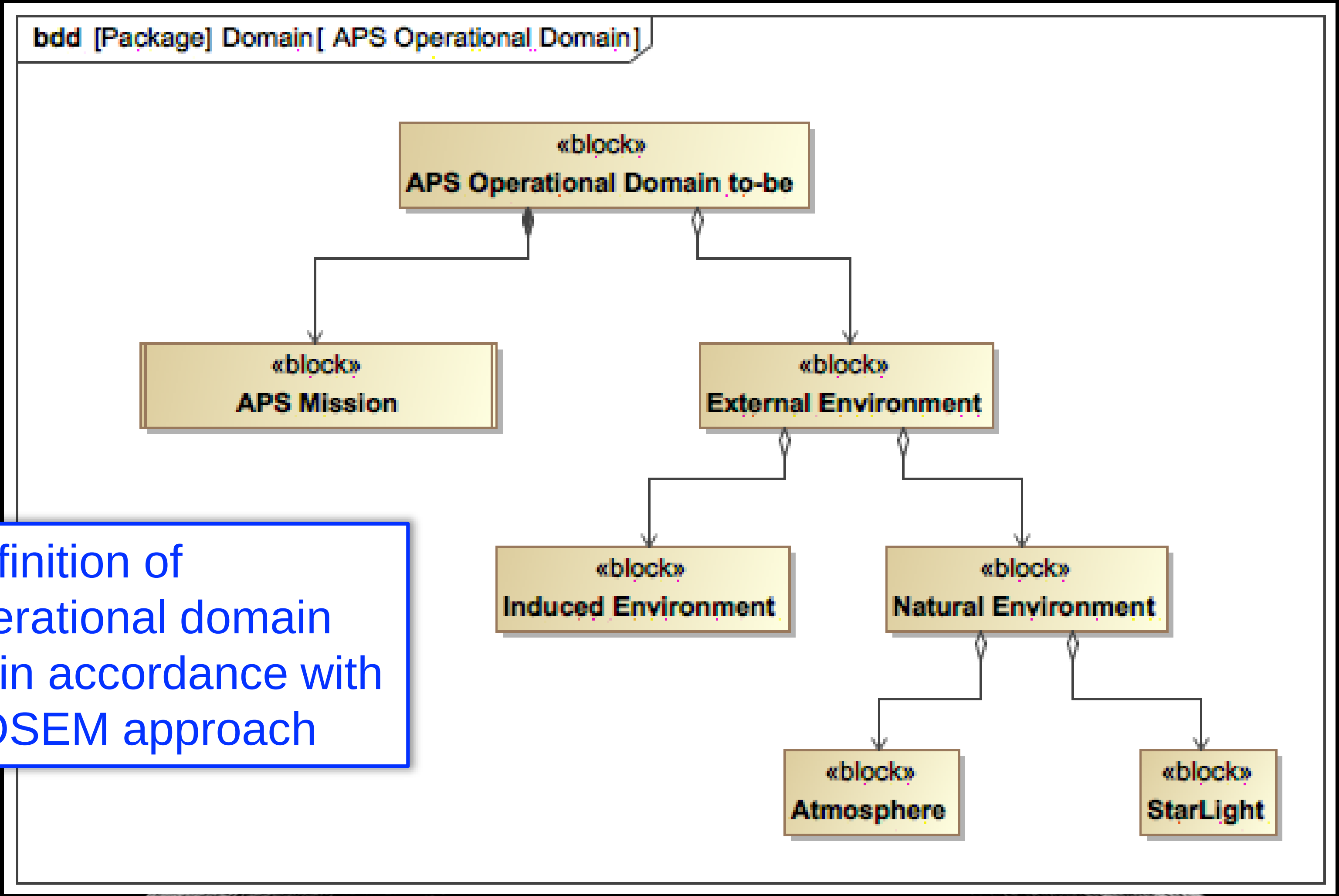
OOSEM abstraction layers





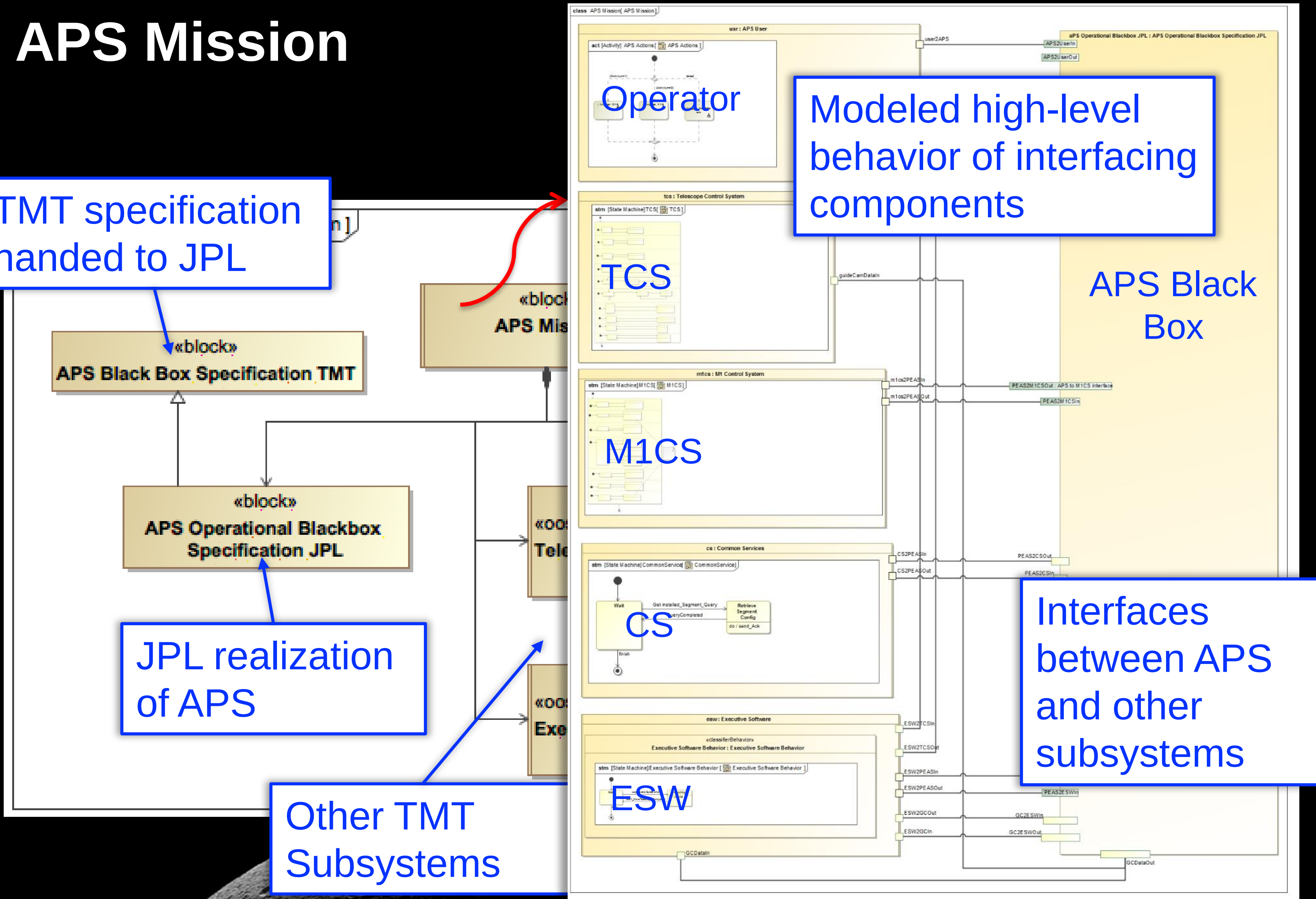
Work breakdown structure

Operational Domain



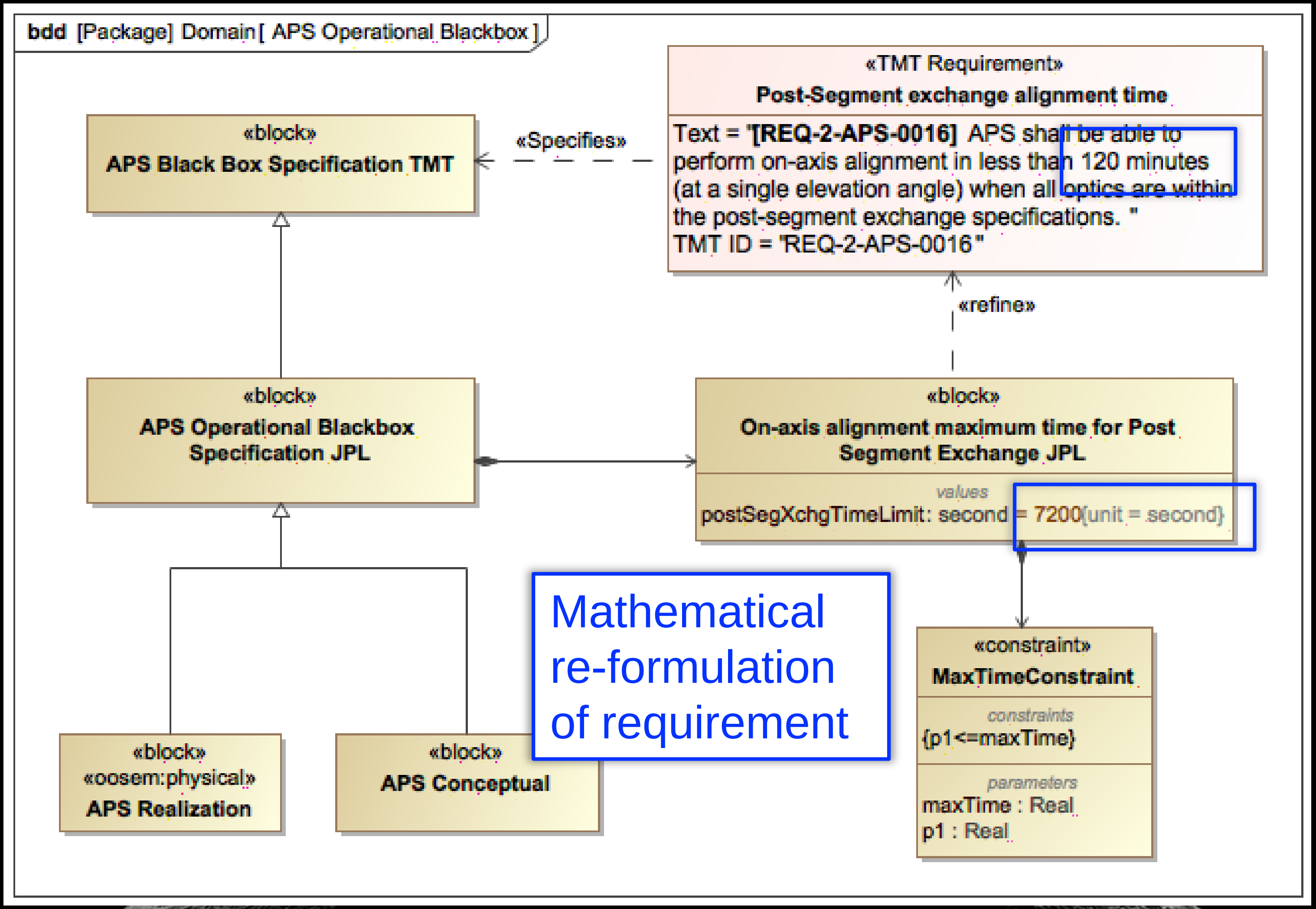
Definition of operational domain as in accordance with OOSEM approach

APS Mission

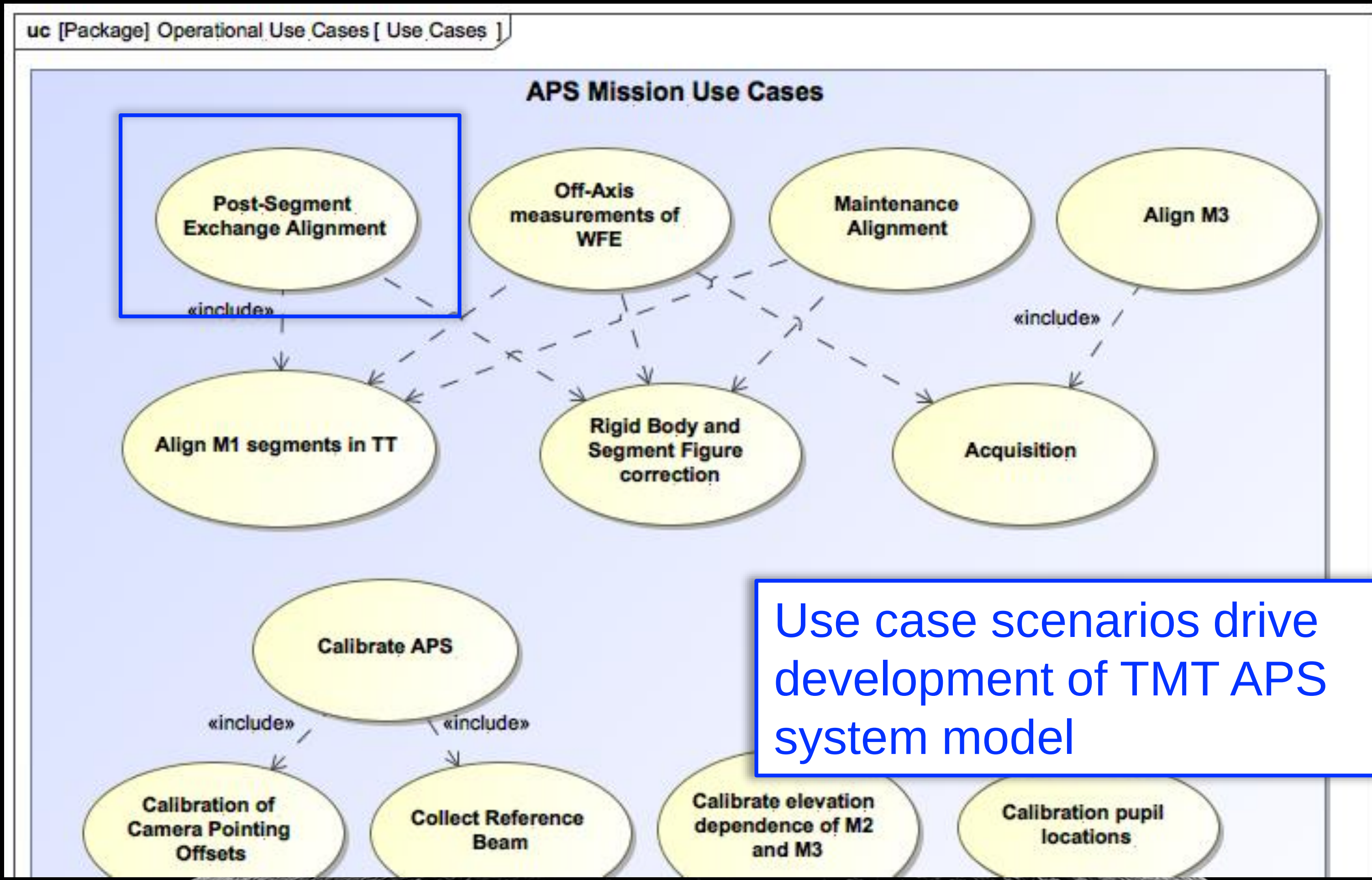


TMT specification handed to JPL
«block»
APS Black Box Specification TMT
«block»
APS Operational Blackbox Specification JPL
JPL realization of APS
Other TMT Subsystems
Operator
TCS
M1CS
CS
ESW
Modeled high-level behavior of interfacing components
APS Black Box
Interfaces between APS and other subsystems

Formalizing Requirements

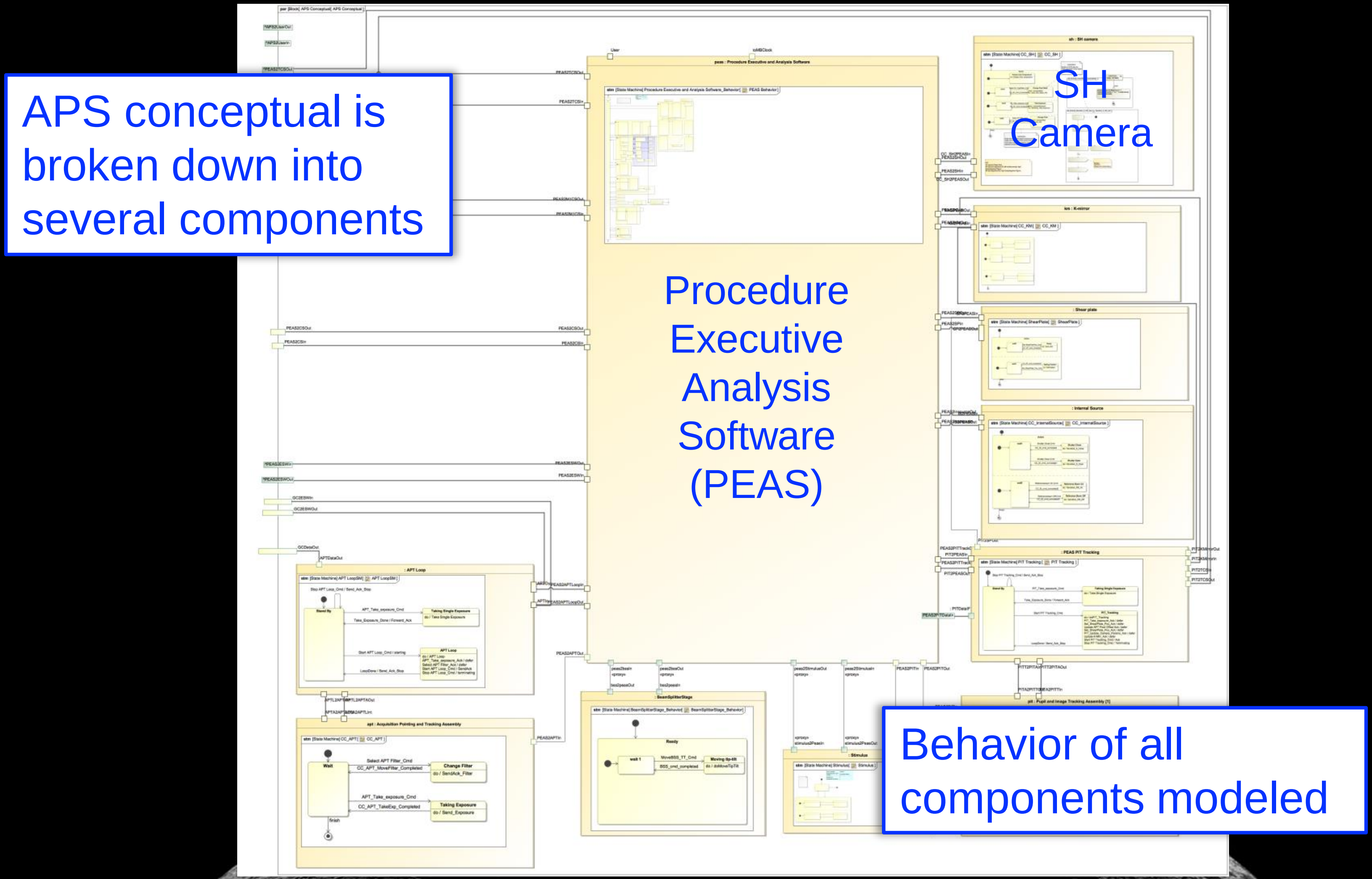


Use Cases

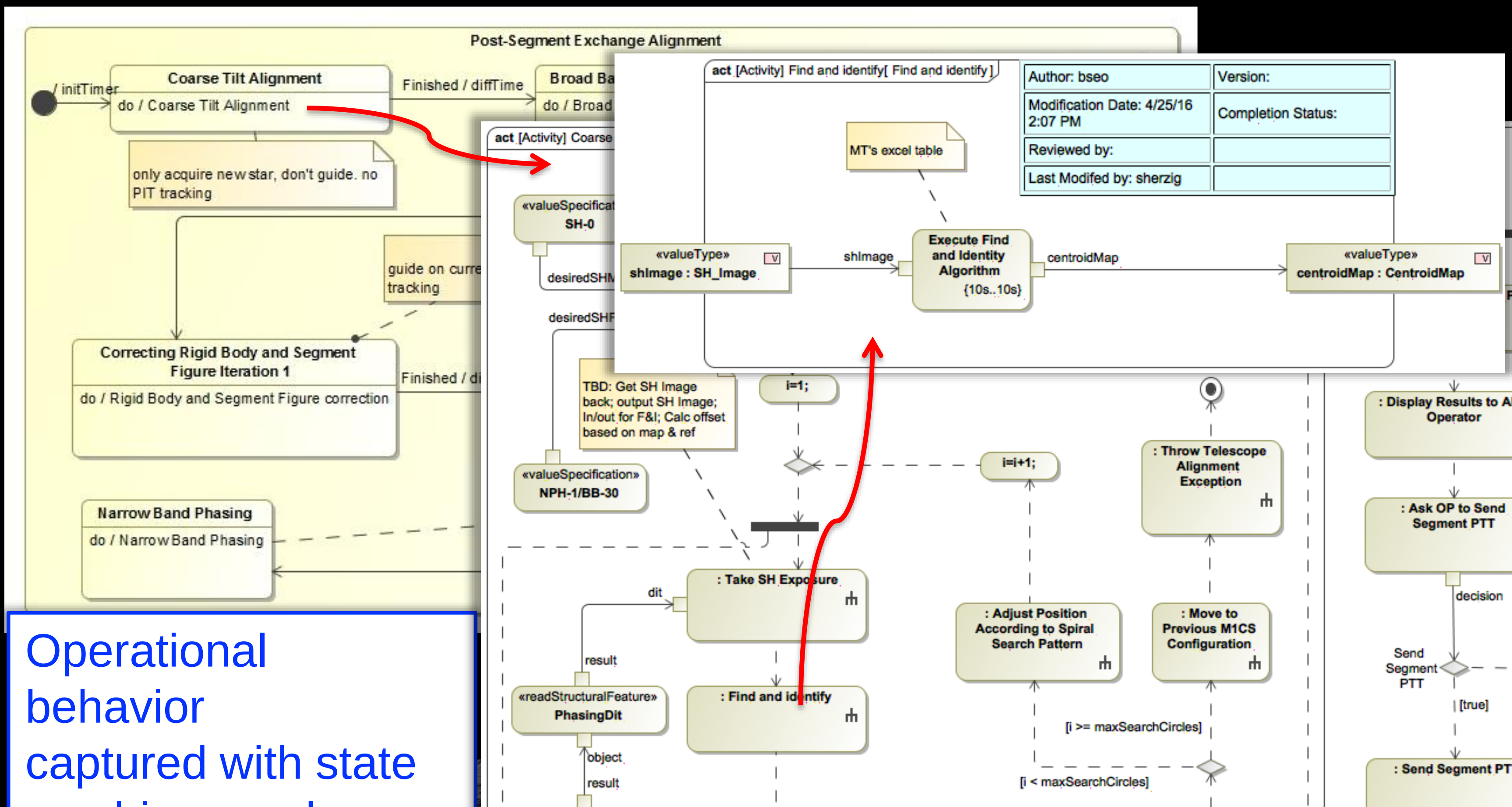


Use case scenarios drive development of TMT APS system model

Conceptual Architecture



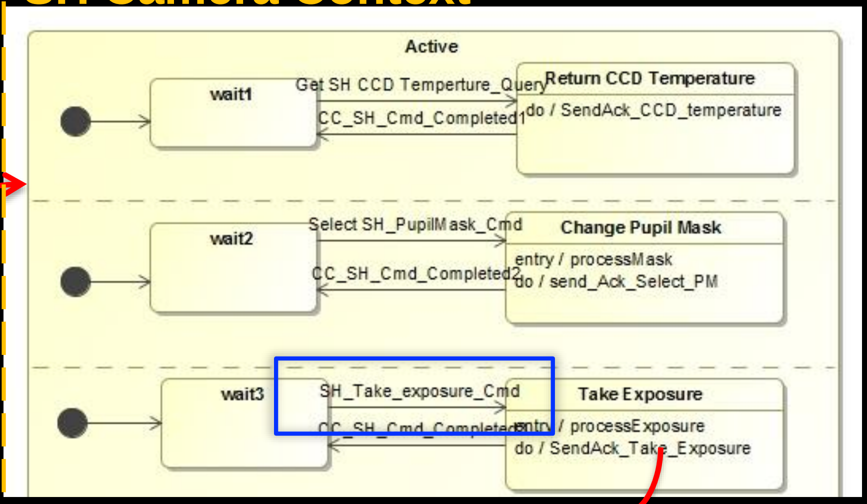
Modeling Behavior



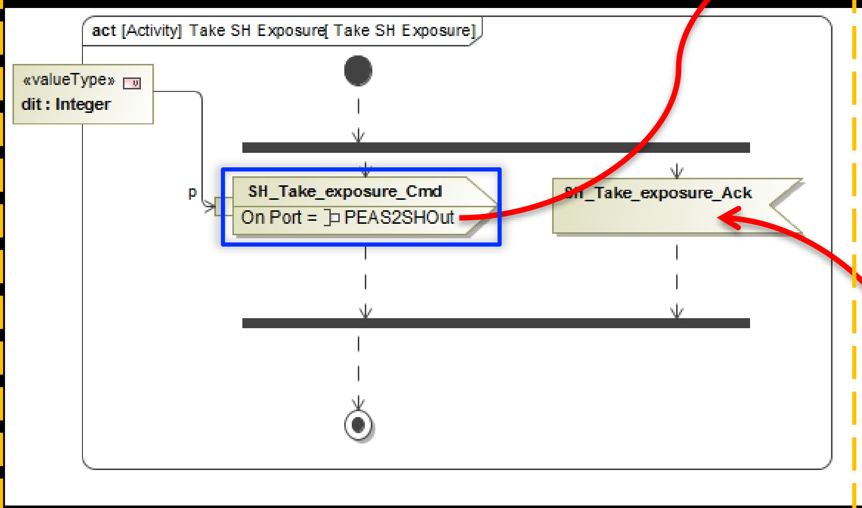
Operational behavior captured with state machines and activity models

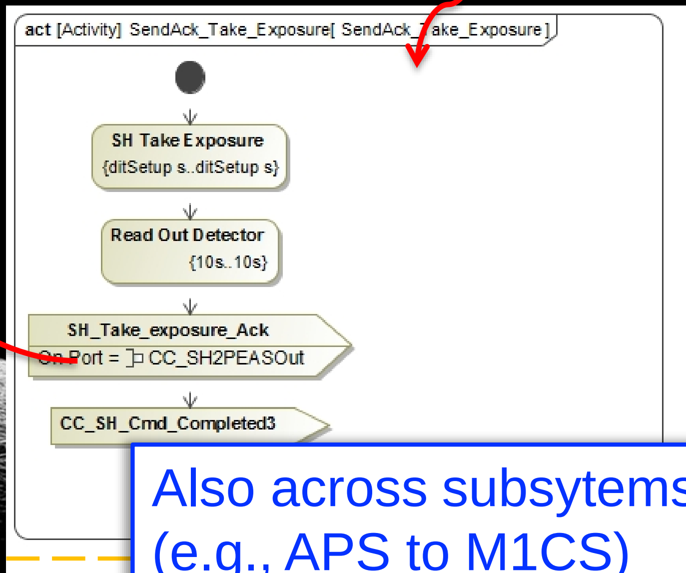
Interactions Between Components

SH Camera Context



PEAS Context

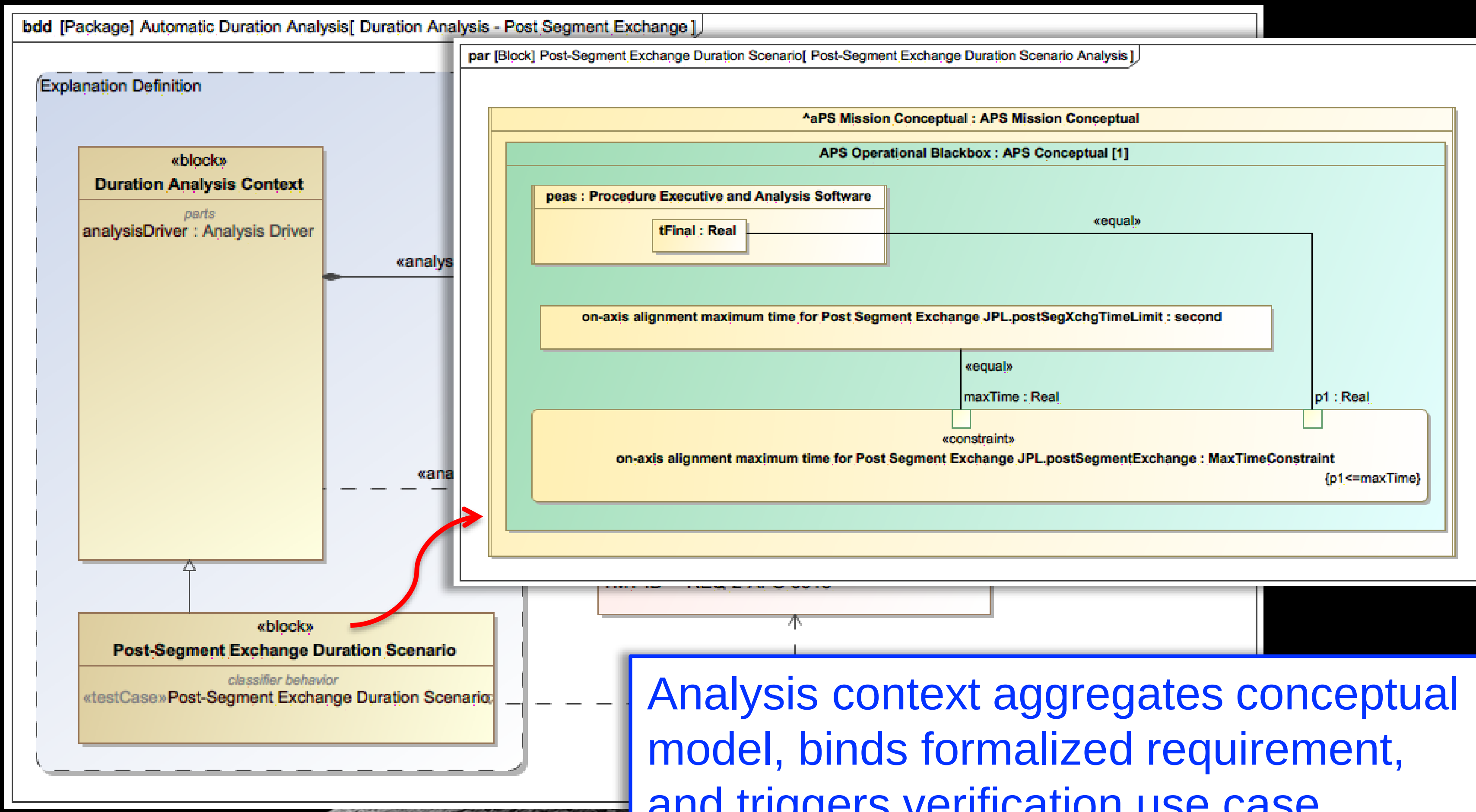




Use of signals sent over ports to simulate a message passing mechanism between components

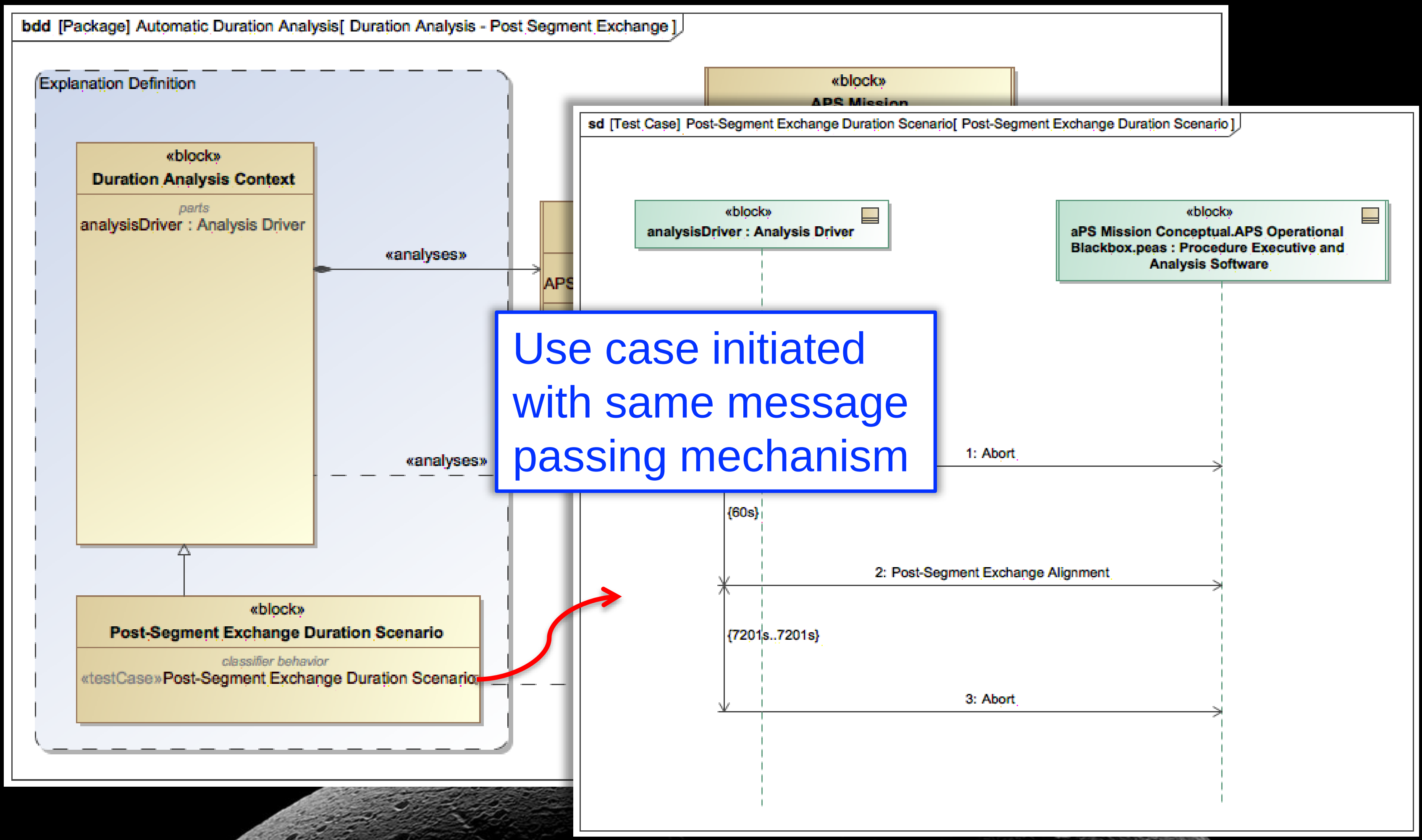
Also across subsytems! (e.g., APS to M1CS)

Verifying Timing Requirements by Simulation

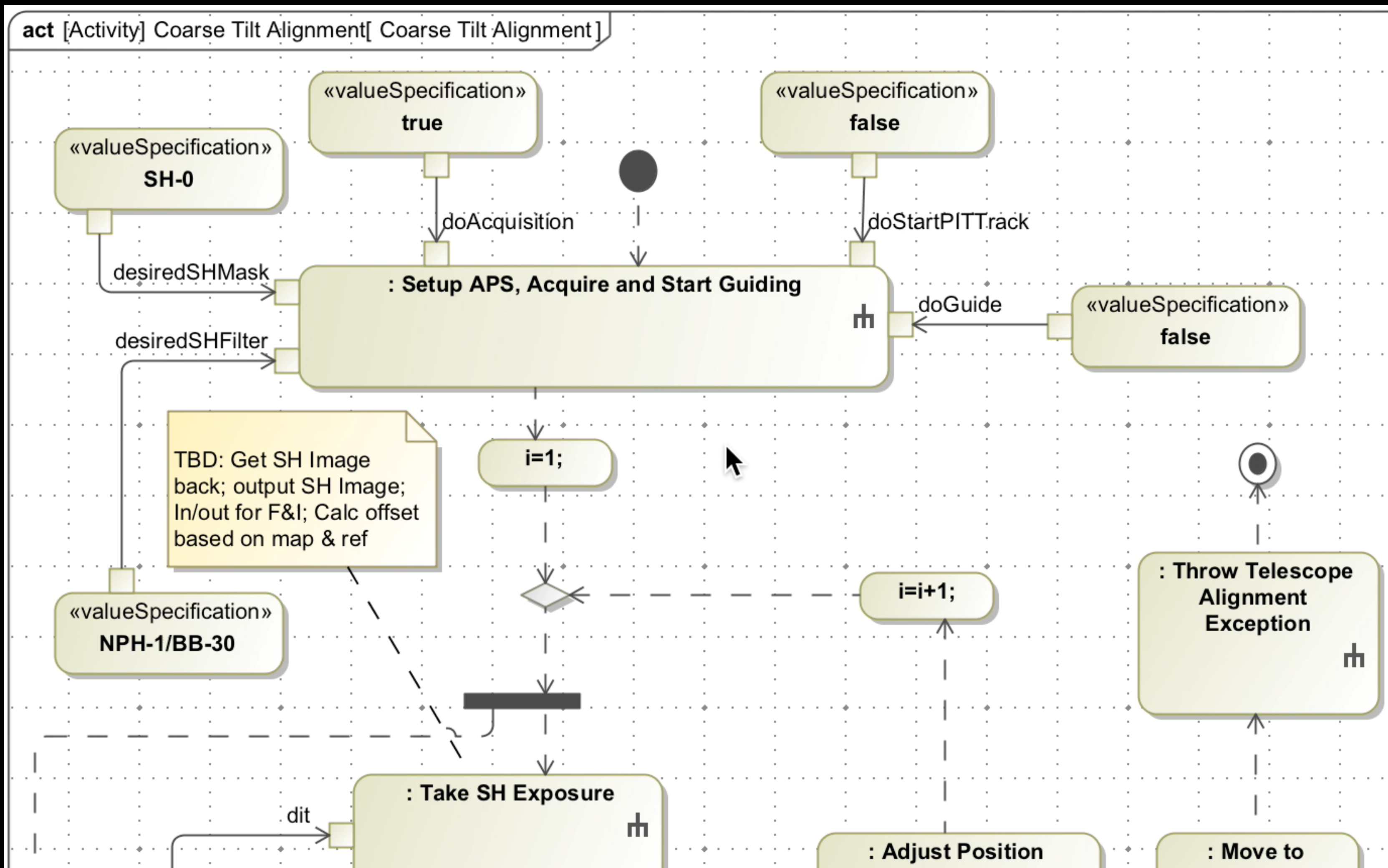


Analysis context aggregates conceptual model, binds formalized requirement, and triggers verification use case

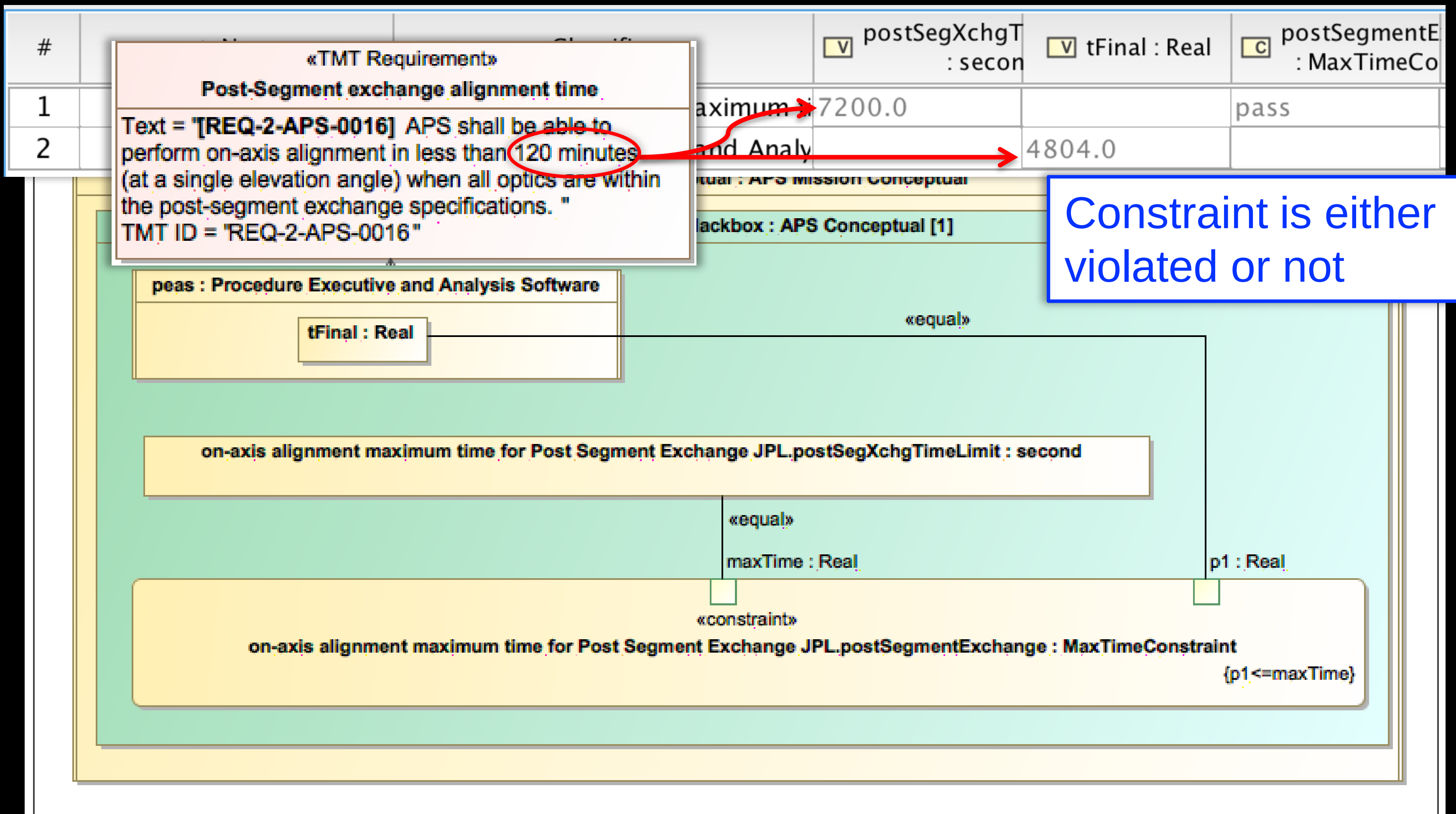
Verifying Timing Requirements by Simulation



Verifying Timing Requirements by Simulation

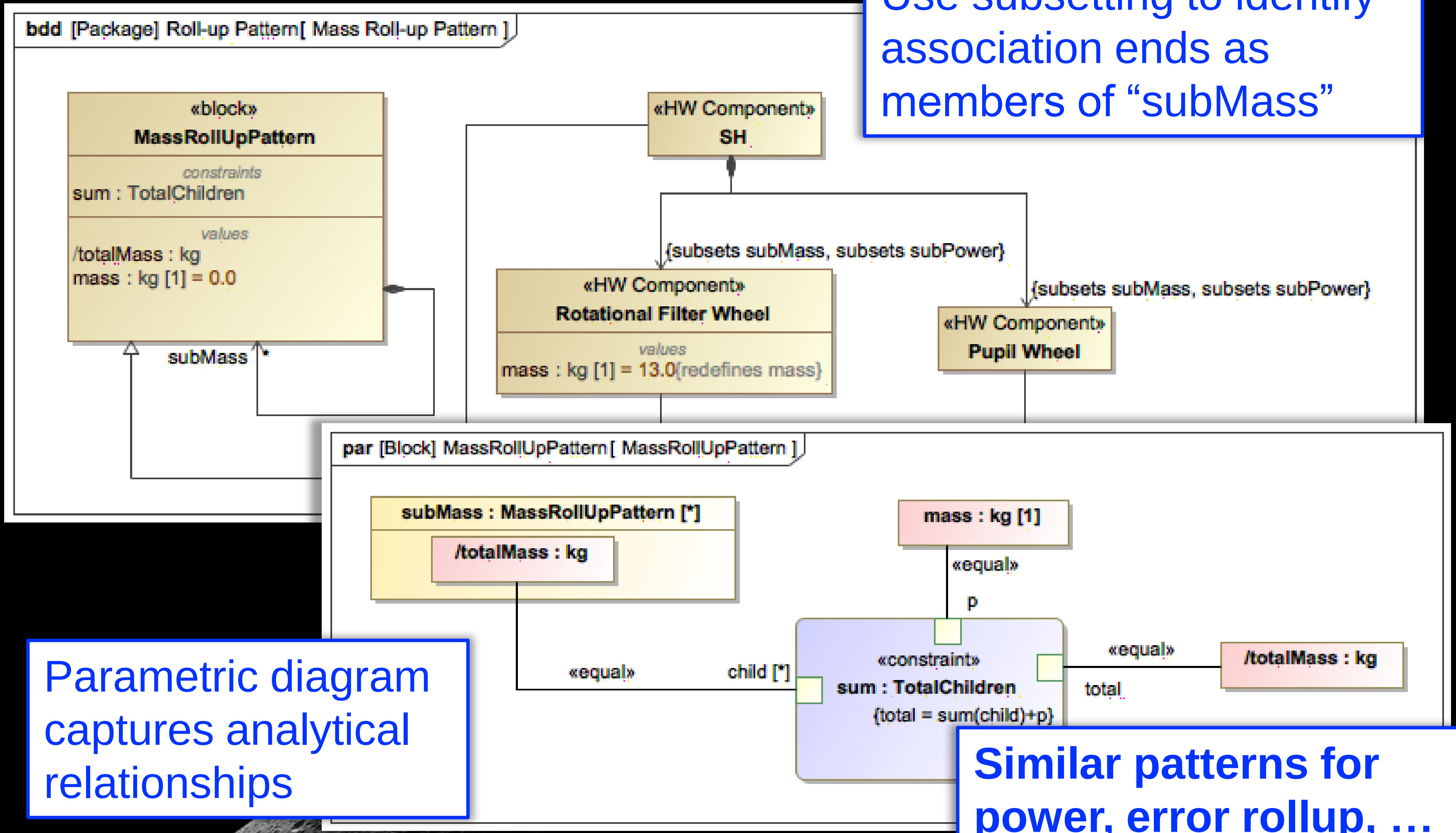


Verifying Timing Requirements by Simulation



“Static” Rollup Analyses – Example: Mass

Use subsetting to identify association ends as members of “subMass”



Parametric diagram captures analytical relationships

Similar patterns for power, error rollup, …

Agenda

- Introduction
- Problem Statement
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

Conclusions and Summary

- JPL is successfully applying Model-Based Engineering over numerous projects
- There has been tremendous progress in tools and methodology in recent years
- The paradigm shift is manifesting in a vibrant open-source community of practitioners from around the world

Acknowledgements

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

The TMT Project gratefully acknowledges the support of the TMT collaborating institutions. They are the Association of Canadian Universities for Research in Astronomy (ACURA), the California Institute of Technology, the University of California, the National Astronomical Observatory of Japan, the National Astronomical Observatories of China and their consortium partners, and the Department of Science and Technology of India and their supported institutes. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the National Research Council of Canada, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Universities for Research in Astronomy (AURA) and the U.S. National Science Foundation.

References

- Karban, R., Jankevičius, N., Elaasar, M. “ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns”, (to appear in the proceedings of INCOSE International Symposium (IS), Edinburgh, Scotland, 2016.)
- Karban R., Dekens F., Herzig S., Elaasar M, Jankevičius N., “Creating systems engineering products with executable models in a model-based engineering environment”, SPIE, Edinburgh, Scotland, 2016
- Karban, R., “Using Executable SysML Models to Generate Systems Engineering Products”, NoMagic World Symposium, Allen, TX, 2016
- Open Source TMT model: https://github.com/Open-MBEE/TMT-SysML-Model
- Open Source Engineering Environment: https://open-mbee.github.io/
- Docgen, View&ViewPoints: https://github.com/Open-MBEE/mdk/tree/mdk-manual/src/main/dist/manual
- JPL Model-Based Systems Engineering Case Study: http://omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:incose_mbse_iw_2017:iw_2017_open_mbee.pdf
- A Practical Guide to SysML, 3rd Edition, Chapter 17 by Friedenthal, Moore, and Steiner
- Zwemer, D., “Connecting SysML with PLM/ALM, CAD, Simulation, Requirements, and Project Management Tools”, May 2016
- https://www.jpl.nasa.gov/spaceimages/

Jet Propulsion Laboratory
California Institute of Technology

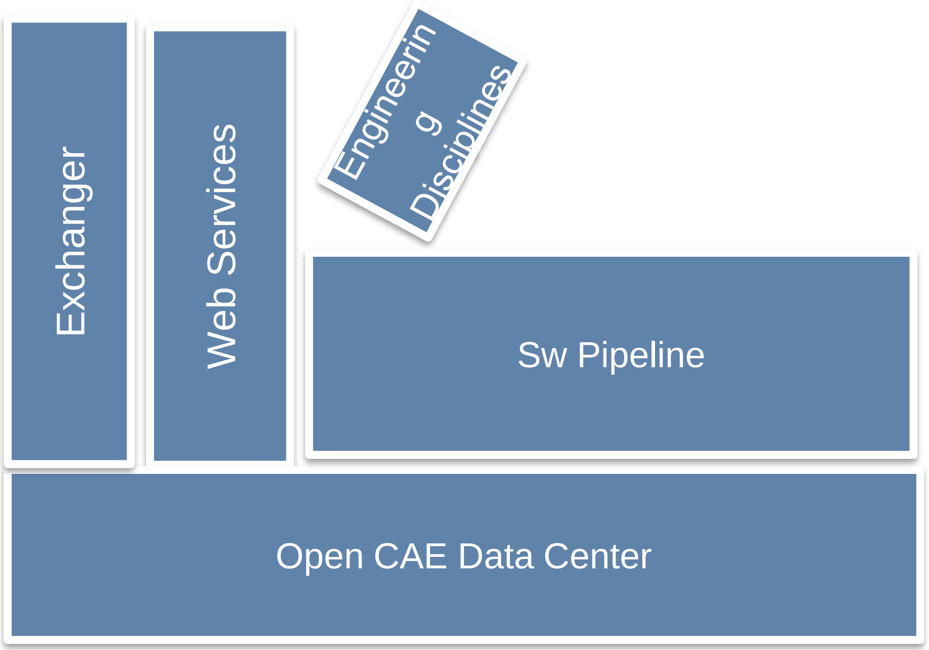
jpl.nasa.gov

BACKUP SLIDES

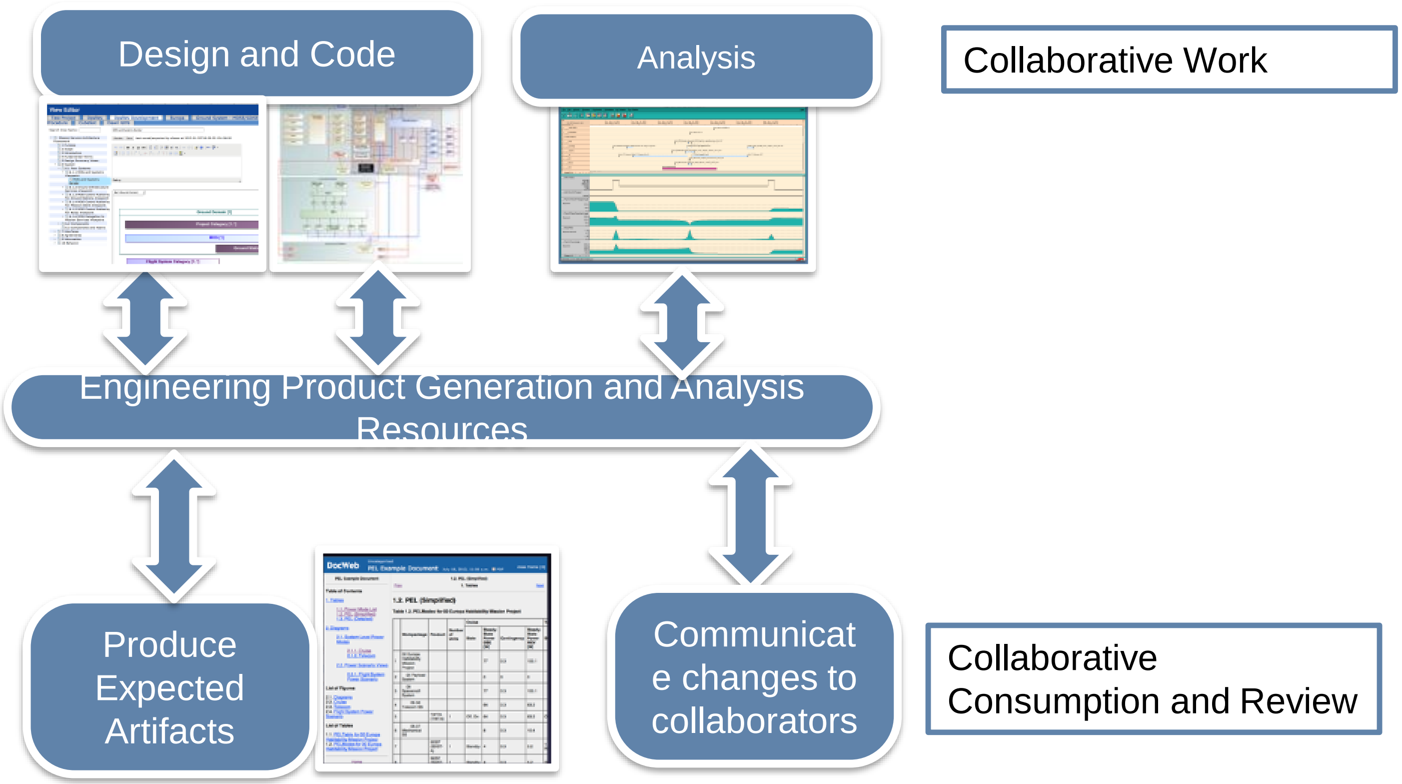
Open CAE

Open CAE Commodity Operations

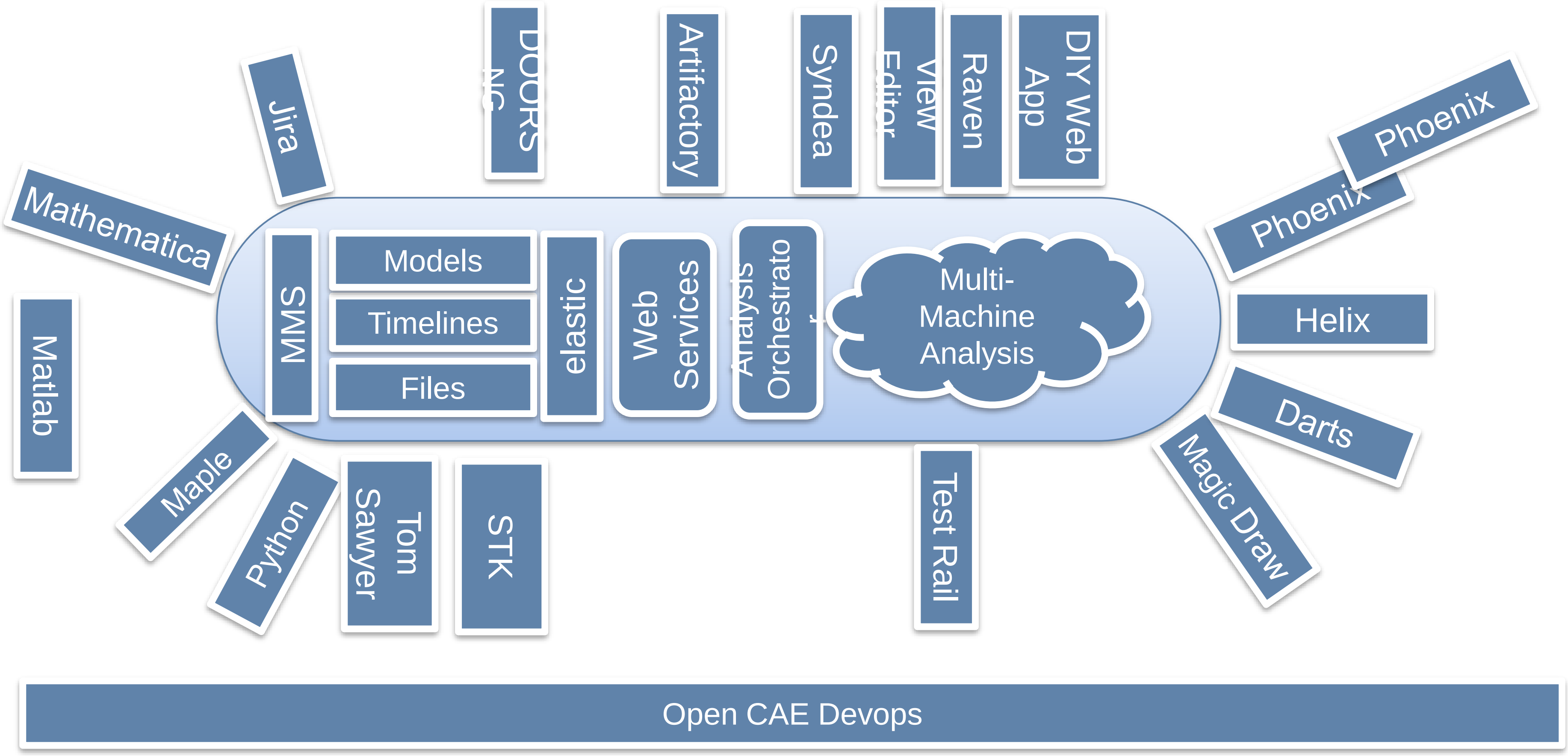
- Unified Experience - Exchanger provides unified experience with web services and apps for CAE users
- Pipline - Software pipeline manages automated provisioning of Engineering Discipline tools, Web Services, Exchanger, and Apps for CAE and Flight Projects consistent with devops architecture
- Data Center provides commodity resources and facilities



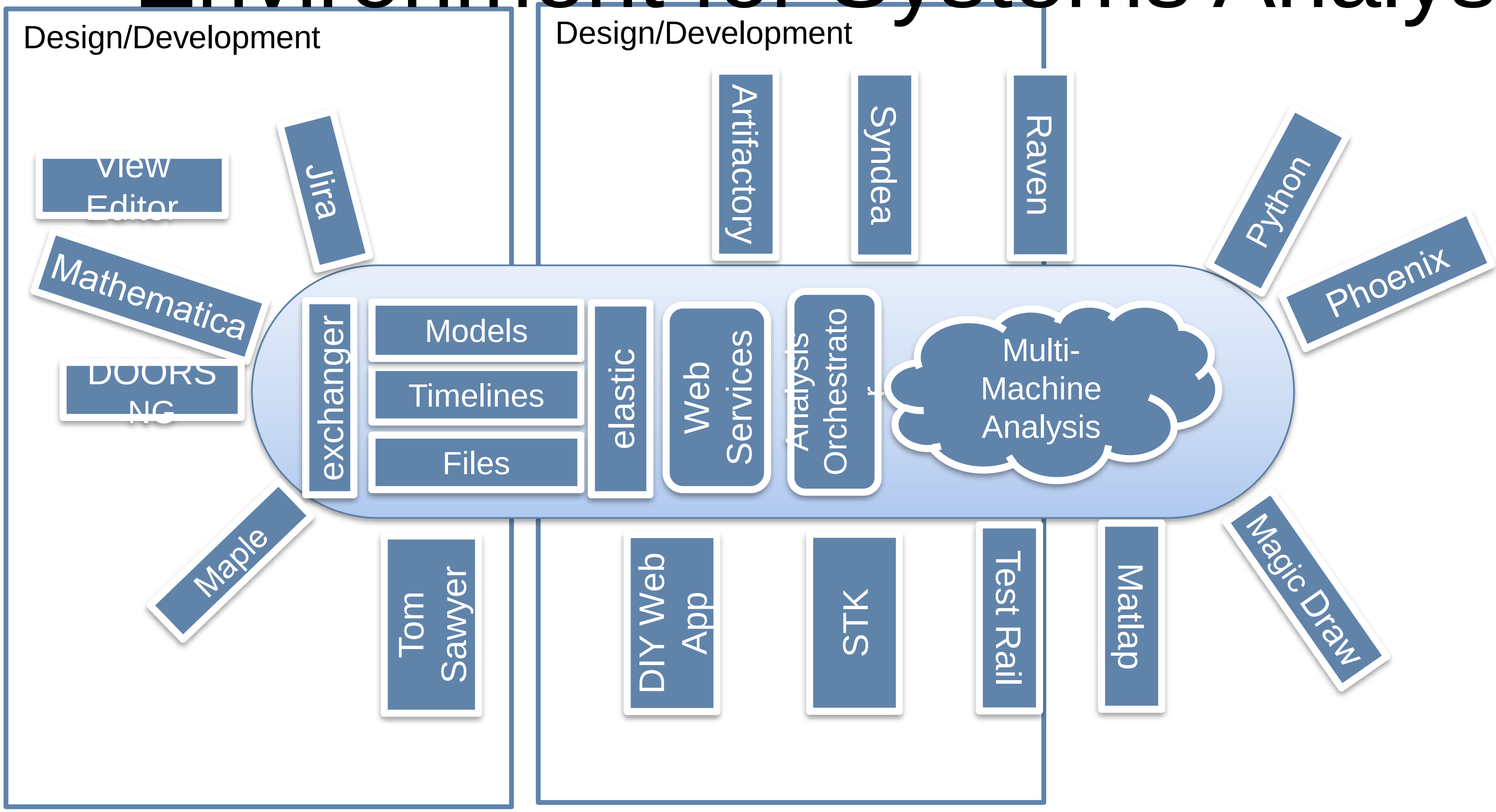
Workflow Scenario



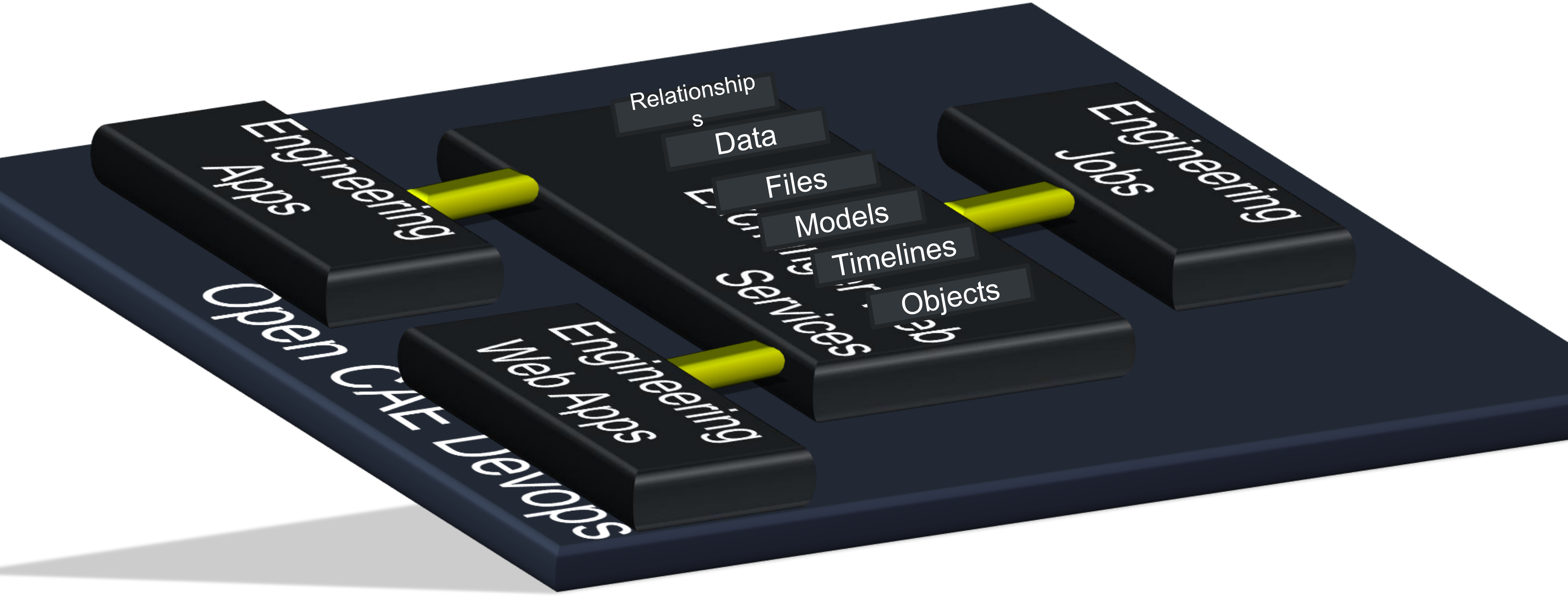
Environment for Systems Analysis



Environment for Systems Analysis



Environment for SE



Model authoring (description)

- Engineering tools
  - Magic draw/data hub/simtk
  - Phoenix Model Center/cloud
  - Syndea
  - View editor
  - iPython
  - Maple/modelica
  - Mathematica/modelica
  - Matlab/Simulink
  - Doors ng
  - Stk
  - raven

analysis

- System analysis – platform for model analysis (PMA)
  - Service for CI-oriented batch/large scale analysis
  - Web service wrappers
  - Configured for cae analysis tools
    - Sim tk/data hub
    - Phoenix
    - Stk
    - Ipython/python/code
    - Matlab, maple, Mathematica
    - Syndea

- Semantic web centric architecture
  - We don’t architect with semantic web as the foundation
  - Web services, models and graphs
  - We don’t consider any language technology or model as dominant – just web services and models
- Domain specific languages, embeddings
  - We have centralized domain specific language support for custom jpl modeling languages
  - Users accomplish this in a variety of non-standard ways focusing more on evolving
- Platform-specific Custom application support
  - We don’t see need for desktop oriented custom software oriented around 1 specific platform
  - We do see a need for custom light weight web apps using a variety of technologies based on the needs
  - We see a broad need for web services access for data manipulation, analysis etc
- We don’t see Engineers working directly in OWL/OML
  - We see Engineers working with Modeling languages but customized to their needs.
  - Executable designs are the strongest drivers in the community of CAE users

Ontology Authoring

- Cameo Concept Modeler provides full ontology modeling and checking
- IMCE Potential
    - OML Modeling
    - OWL Modeling and Model-Checking
    - Profile Generation

Design Authoring

- MMS can store semantic models in EMF
  - Could add RDF level access
  - AWS Neptune has RDF built in - planned
  - Has API for analysis extraction etc
  - Element level versioning
  - Branching capability
  - Visualizations with Tom Sawyer, D3 and open framework for more
  - Technical search
- Commercial Authoring Tools have rich integration
  - Don't require OML adapter – benefits of such adapters are unclear
    - Large number of commercial integrations exist and are expanding every day
    - The commercial integrations a detailed and polished
  - Many IMCE embeddings in CAE tools break tools full capability
  - Concept Modeler is available for ontology modeling with rules checking
- Commercial Ontology Authoring
  - Cameo Concept Modeler provides full ontology modeling and checking
  - Fully compatible with OML
- Integrated Analysis
  - Jenkins based batch automation
  - Fully integrated Analysis for co-simulation, flat-parameter numerical analysis, web-services with batch capability
- Rich Document Generation and Reporting with Interactive Authoring
  - Ve docgen
  - Full web based interactivity including authoring

Integration and Integrated Analysis

- Commercial Authoring Tools have rich integration
  - Don’t require OML adapter – benefits of such adapters are unclear
    - Large number of commercial integrations exist and are expanding every day
    - The commercial integrations a detailed and polished
  - Many IMCE embeddings in CAE tools break tools full capability
  - Concept Modeler is available for ontology modeling with rules checking
- Integrated Analysis
  - Jenkins based batch automation
  - Fully integrated Analysis for co-simulation, flat-parameter numerical analysis, web-services with batch capability

Test VnV Process

- MMS can store semantic models in EMF
  - Could add RDF level access
  - AWS Neptune has RDF built in - planned
  - Has API for analysis extraction etc
  - Element level versioning
  - Branching capability
  - Visualizations with Tom Sawyer, D3 and open framework for more
  - Technical search
- Commercial Authoring Tools have rich integration
  - Don't require OML adapter – benefits of such adapters are unclear
    - Large number of commercial integrations exist and are expanding every day
    - The commercial integrations a detailed and polished
  - Many IMCE embeddings in CAE tools break tools full capability
  - Concept Modeler is available for ontology modeling with rules checking
- Commercial Ontology Authoring
  - Cameo Concept Modeler provides full ontology modeling and checking
  - Fully compatible with OML
- Integrated Analysis
  - Jenkins based batch automation
  - Fully integrated Analysis for co-simulation, flat-parameter numerical analysis, web-services with batch capability
- Rich Document Generation and Reporting with Interactive Authoring
  - Ve docgen
  - Full web based interactivity including authoring

Integrated Data Resources

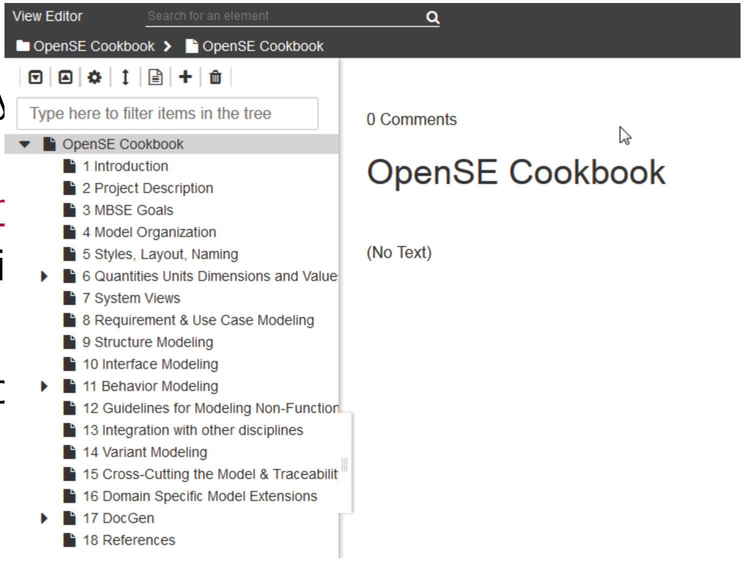
- MMS can store semantic models in EMF
  - Could add RDF level access
  - AWS Neptune has RDF built in - planned
  - Has API for analysis extraction etc
  - Element level versioning
  - Branching capability
  - Visualizations with Tom Sawyer, D3 and open framework for more
  - Technical search
- Commercial Authoring Tools have rich integration
  - Don’t require OML adapter – benefits of such adapters are unclear
    - Large number of commercial integrations exist and are expanding every day
    - The commercial integrations a detailed and polished
  - Many IMCE embeddings in CAE tools break tools full capability
  - Concept Modeler is available for ontology modeling with rules checking
- Commercial Ontology Authoring
  - Cameo Concept Modeler provides full ontology modeling and checking
  - Fully compatible with OML
- Integrated Analysis
  - Jenkins based batch automation
  - Fully integrated Analysis for co-simulation, flat-parameter numerical analysis, web-services with batch capability
- Rich Document Generation and Reporting with Interactive Authoring
  - Ve docgen
  - Full web based interactivity including authoring

How Could CAE be Used

- MMS can store semantic models in EMF
  - Could add RDF level access
  - AWS Neptune has RDF built in - planned
  - Has API for analysis extraction etc
  - Element level versioning
  - Branching capability
  - Visualizations with Tom Sawyer, D3 and open framework for more
  - Technical search
- Commercial Authoring Tools have rich integration
  - Don't require OML adapter – benefits of such adapters are unclear
    - Large number of commercial integrations exist and are expanding every day
    - The commercial integrations a detailed and polished
  - Many IMCE embeddings in CAE tools break tools full capability
  - Concept Modeler is available for ontology modeling with rules checking
- Commercial Ontology Authoring
  - Cameo Concept Modeler provides full ontology modeling and checking
  - Fully compatible with OML
- Integrated Analysis
  - Jenkins based batch automation
  - Fully integrated Analysis for co-simulation, flat-parameter numerical analysis, web-services with batch capability
- Rich Document Generation and Reporting with Interactive Authoring
  - Ve docgen
  - Full web based interactivity including authoring

INCOSE Telescope Challenge team - OpenSE Cookbook

- Challenge Team: Demonstrate benefits of MBSE with SysML as the basis for integrated engineering and management of complex systems
  - Optimization, standardization, automation
  - Better system understanding through simulations and analyses
  - Early efforts go a long way (reduced risk/cost, expand knowledge)
  - http://www.omgwiki.org/MBSE/doku.php?id=mbse:telescope
- Cookbook: New revision of OpenSE Cookbook for MBSE with SysML
  - Practitioner oriented
  - Best practices to support common SE tasks
  - Patterns and practices for model construction and analy
  - Express system concepts to diverse stakeholders
  - Current revision: http://mbse.gfse.de/documents/faq.htr
- INCOSE SEBoK TMT Case Study for Fall 2017 publi
- Ongoing collaboration in telescope community
- Using TMT as reference model for OpenSE Cookboc
- Provide input to SysML 2.0 RfP
- Open Source



Executable Models

- Most SysML models today are created for documentation purposes
  - The focus is on syntax and notation
- Some SysML models are created to gain system understanding, explore and validate desirable or undesirable behaviors of a system
  - The focus is on semantics

Object Oriented System Engineering Method

Defines the architecture in terms of:

- Domain: the context of the solution
  - Enterprise: the ecosystem of the solution
    - System of Interest: the solution being specified
      - Black Box: externally visible specification
      - Conceptual: white box functional specification
      - Physical: white box realization specification

Model Execution

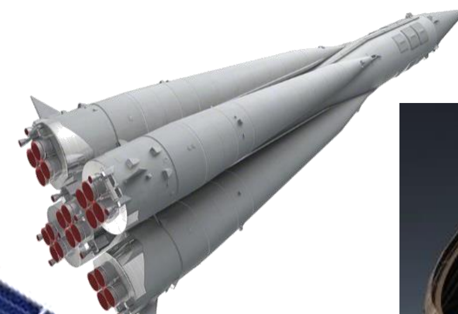
- Executable SysML models are defined with a subset of the language with well defined execution semantics
  - The subset is called Foundational UML (fUML)
  - SysML inherits the fUML subset from UML
- SysML models are executed with the help of an execution, or simulation engine
  - Ex.: NoMagic’s Cameo Simulation Toolkit (CST)

Cameo Simulation Toolkit (CST)

- A plugin to MagicDraw SysML modeling tool
- A simulation platform based on fUML and plugs in additional execution engines
  - State Chart XML (SCXML)
  - Scripting for the Java Platform (JSR 223)
  - Precise Semantics of Composite Structures (PSCS)
  - Precise Semantics for State Machines (PSSM)

Complexity of TMT

- The Thirty Meter Telescope (TMT) is no different than other complex systems of systems

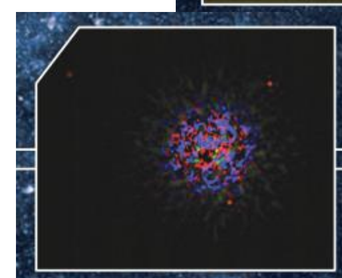


- We still need to apply core SE processes
- Difference: telescope community is historically unfamiliar with formal Systems Engineering

TMT Key Science

- Nature and composition of the Universe
- Formation of the first stars and galaxies
- Evolution of galaxies
- Relationship between black holes and their galaxies
- Formation of stars and planets
- Nature of extra-solar planets
- Potential of life elsewhere in the Universe
- Unforeseen discoveries…



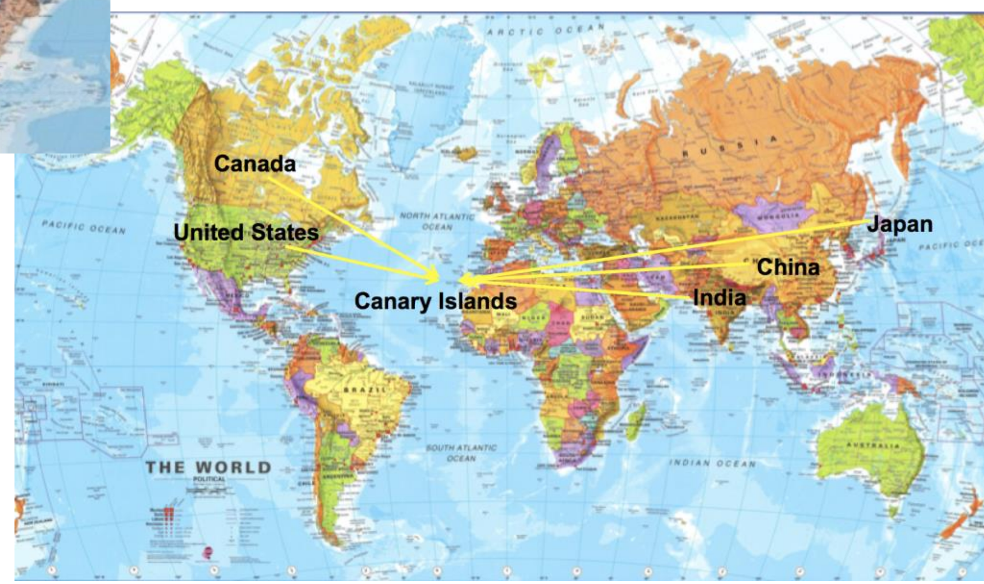



TMT Project

- TMT Project formed in 2004
- TMT international partnership grew
  - US (Caltech & UC), Canada, China, India, Japan
- 2004 – 2008 site studies
  - Chile, Mexico, Hawaii
- Mauna Kea, Hawaii selected in 2009
- 2014 start of TMT Construction Phase

TMT Site

- Preferred site:
  - Mauna Kea on the Big Island of Hawaii, United States

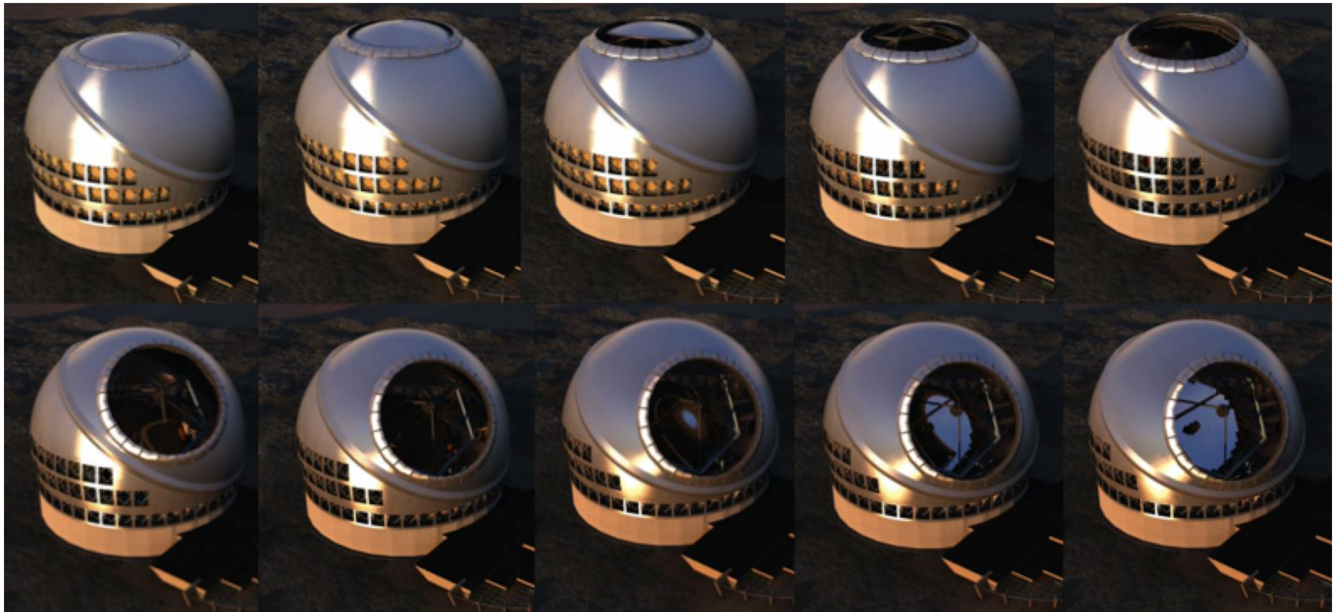


- Alternate site:
  - Observatorio del Roque de los Muchachos (ORM) on La Palma in the Canary Islands, Spain

TMT Enclosure

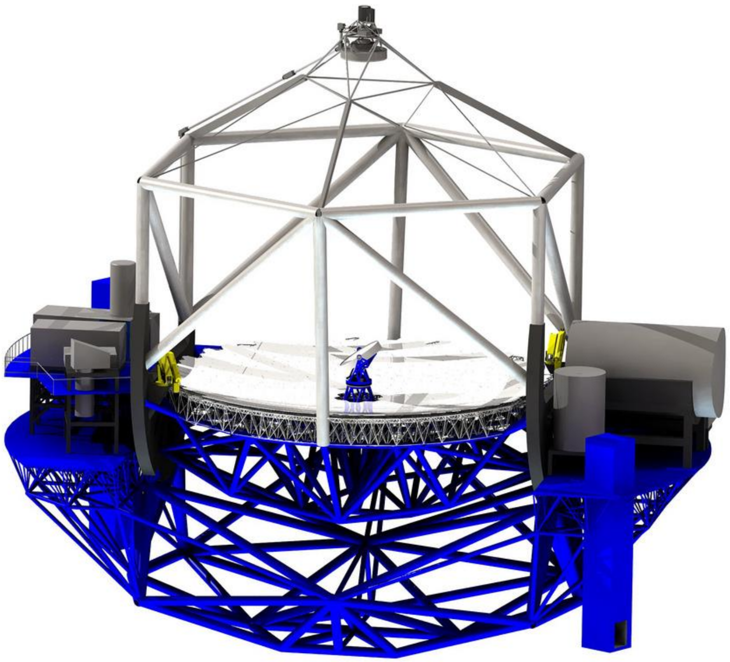
- Calotte design
- Azimuth rotation on fixed base ring
- Rotation of cap structure on tilted bearing ring

- Aerodynamic design minimizes degradation image quality due to air turbulence and thermal influences
  - Smooth exterior
  - Minimal size aperture
  - Aperture flaps
  - Ventilation doors



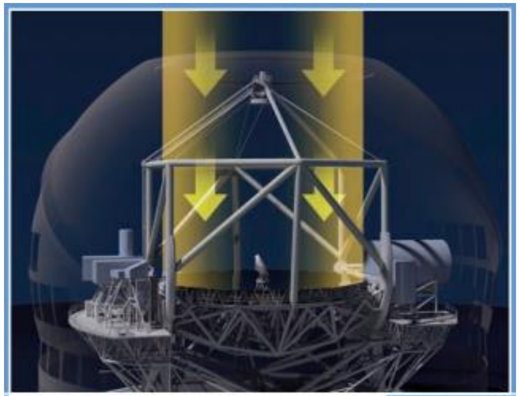
TMT Structure

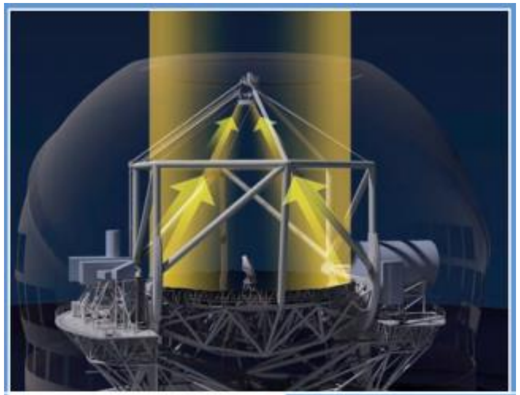
- Elevation structure
  - Mounting support for optics and laser guide star facility
- Azimuth structure
  - Supports elevation structure and 2 large Nasmyth platforms for instruments and AO systems
- Elevators, stairs, walkways, and all utility lines

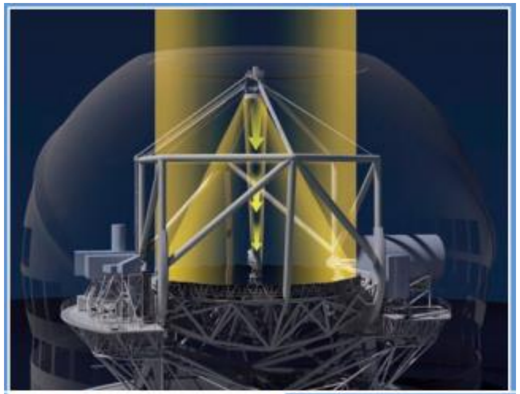


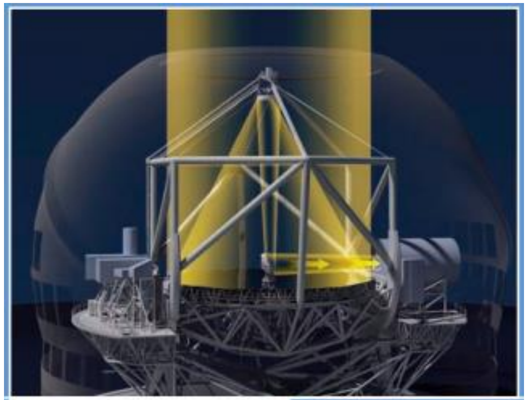
TMT Optics

- 3x larger, 9x more powerful than today's best telescopes
- Ritchey-Chrétien design
- Segmented primary mirror (M1)
  - 492 segments, < 2 m across
  - Collects/concentrates light
- Secondary mirror (M2)
  - Works with M1 to form well-corrected focus
- Tertiary mirror (M3)
  - Steers light to adaptive optics system and science instruments on Nasmyth platforms

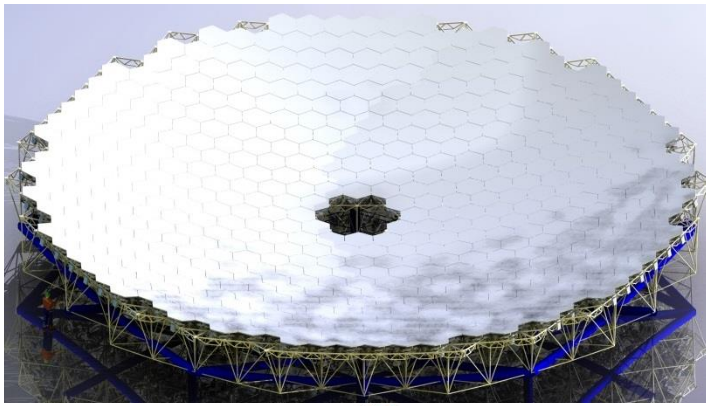

Mirror Path 1

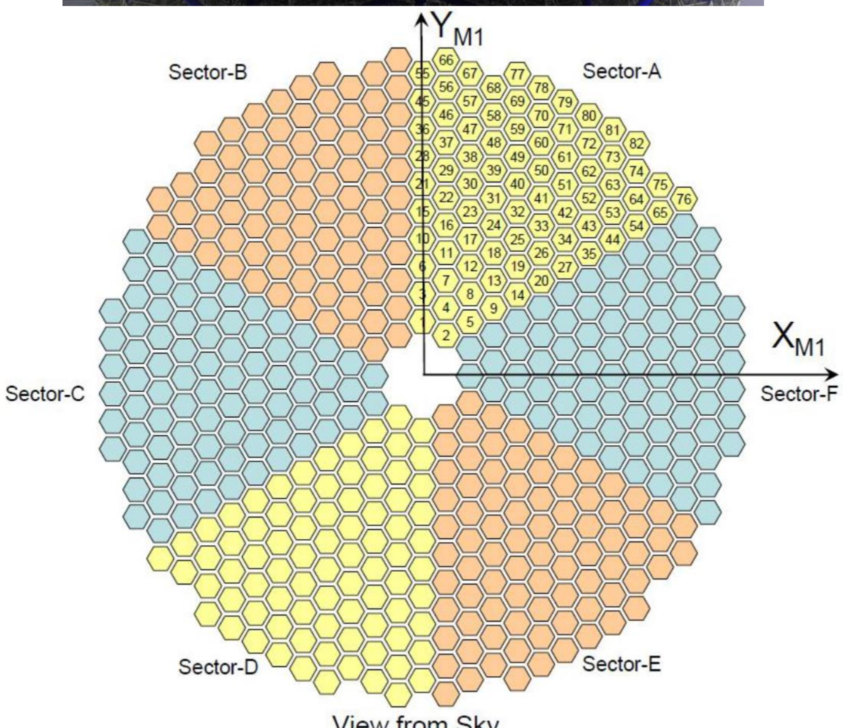

Mirror Path 2


Mirror Path 3


Mirror Path 4

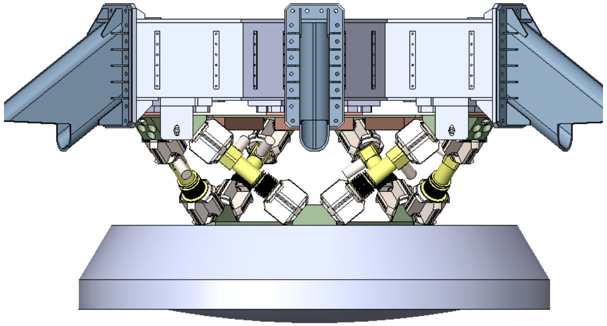
Primary Mirror (M1)





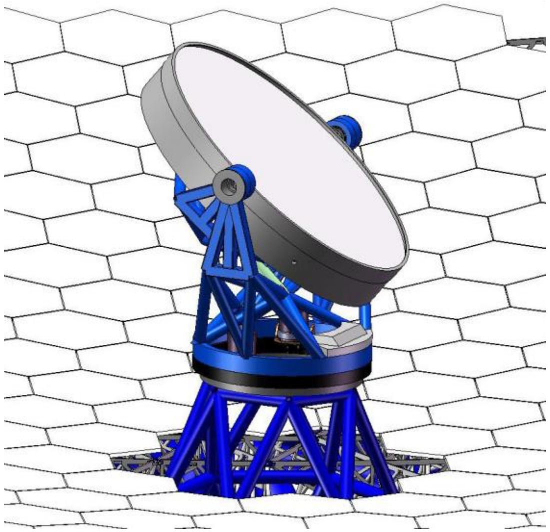
- Segmented primary mirror
  - 492 hexagonal segments
  - 1.44 m across corners
  - 2.5 mm gaps (0.1 in, 0.6% lost area)
  - Thin glass (~2 in) reduces mass and thermal inertia
- Reduces difficulties:
  - Fabrication
  - Testing
  - Transportation
- Reduces risks:
  - Breakage of single segment is less catastrophic
- Moderate cost and complexity

Secondary and Tertiary Mirrors (M2 and M3)



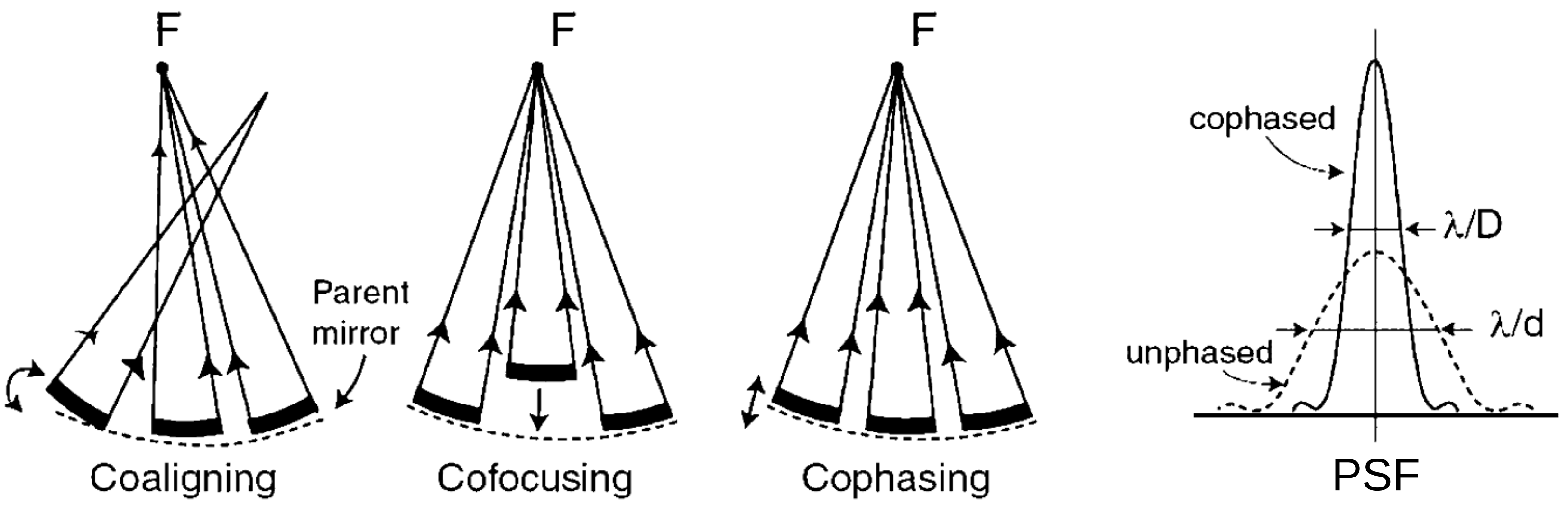
- Secondary Mirror (M2)
  - 3.1 m convex hyperboloid mirror
  - Mounted to telescope top end

- Tertiary Mirror (M3)
  - 2.5 m x 3.5 m flat steerable mirror
  - Rotates and tilts to deliver image to instruments on Nasmyth platforms
  - China (CIOMP) is responsible for design and fabrication



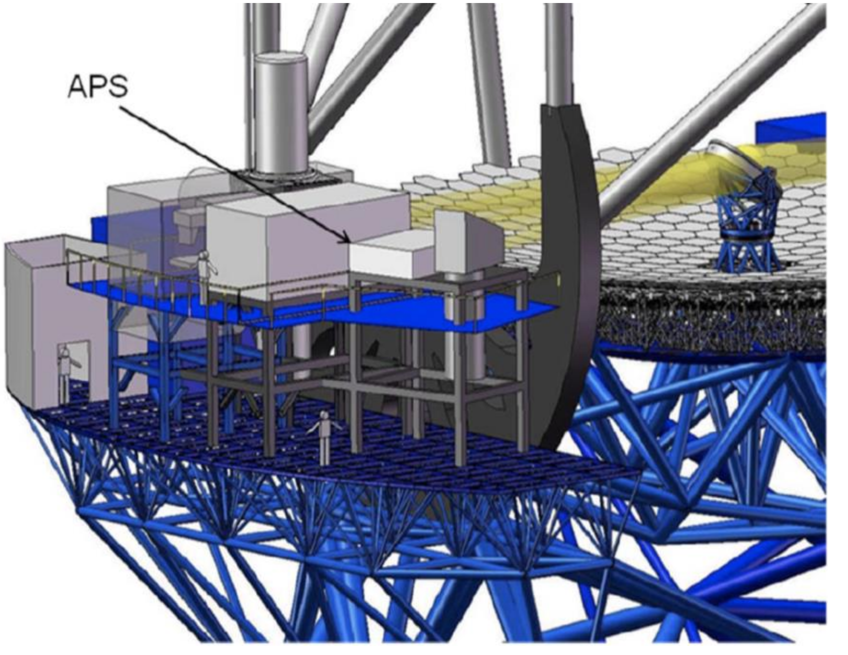
Segmented Mirror Control

- Segmented M1 must perform like a single, smooth mirror to provide optimal image quality
  - **Coaligning**: stacking images produced by each segment to form single image
  - **Cofocusing**: focal lengths of individual segments are equal
  - **Cophasing**: no discontinuities between edges of neighboring segments
- If not phased, image quality = that of individual segment



Alignment and Phasing System (APS)

- Alignment and diagnostic instrument located on a Nasmyth platform
- Modified Shack-Hartmann wavefront sensor
- Responsible for pre-adaptive optics wavefront quality
- Uses starlight to measure wavefront errors and determine commands to send for aligning optics



Typical Analysis Activities Using ESEM

- Capture **operational use cases** with estimated durations of actions, e.g.
  - Post segment-exchange alignment: requirement: 2h; CBE 1h19m
- Capture **power and mass characteristics** of components
- Identify **involved subsystems**, e.g. Telescope Control System (TCS), M1 Control System (M1CS)
- Identify **interfaces and interactions** among subsystems
- **Analyze** associated scenarios
- Automatically **verify system requirements** are satisfied
- Derive requirements for TMT subsystems
- Develop/refine timing requirements for algorithms, internal and external interface commands
- Monte Carlo simulation of expected timings and variants for operational scenarios

MBSE: TMT Application

- Why MBSE?
  - Emphasizes rigor and precision, best practices
  - Helps manage complexity
  - Horizontal (life cycle) and vertical (multiple domain) integration
- TMT SysML model
  - Created to better understand and communicate complex system behavior
  - Executable SysML model to **capture** requirements, use cases, system decomposition, subsystem relationships
  - **Analyze** system design against power, mass, duration requirements
  - **Produce** engineering documents (ICDs, etc.)
  - Use standard language and techniques (**communication**)

MBSE: TMT Application

- TMT SysML Model does not model the entire telescope
- Main objective is to model operational scenarios and demonstrate that requirements are satisfied by the design
- Motivator for TMT MBSE = optimization
- Ex: JPL modeling of APS subsystem
  - Use Case: Post segment-exchange alignment, 2h requirement
  - Component characteristics (power, mass)
  - Relationships (TCS, M1CS)
- Ex: Monte Carlo simulations for acquisition and slew time
  - To minimize loss of observing time, TMT should be able to move from one target to another and acquire it in 3 min or less

Solution: Hybrid Approach

- Traditional SE
  - Clear, defined deliverables
  - Easily accessible
  - Shallow learning curve
  - Simple traceability
- MBSE
  - Understanding behaviors of a system
  - “Rich” capability to represent complex systems

Exploit the advantages of each approach

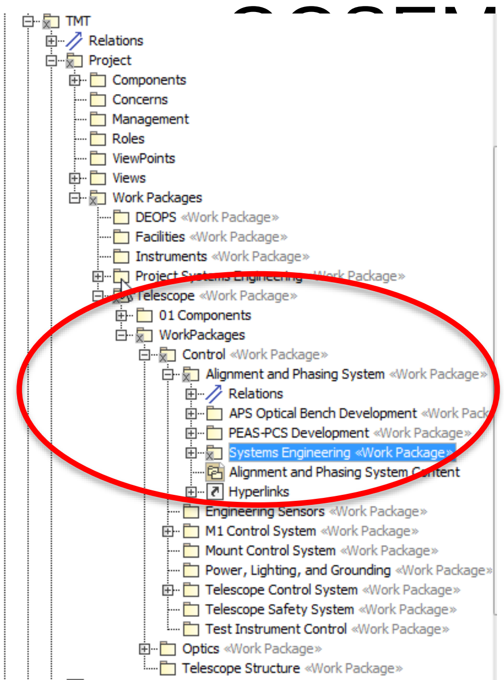
TMT MBSE Objectives

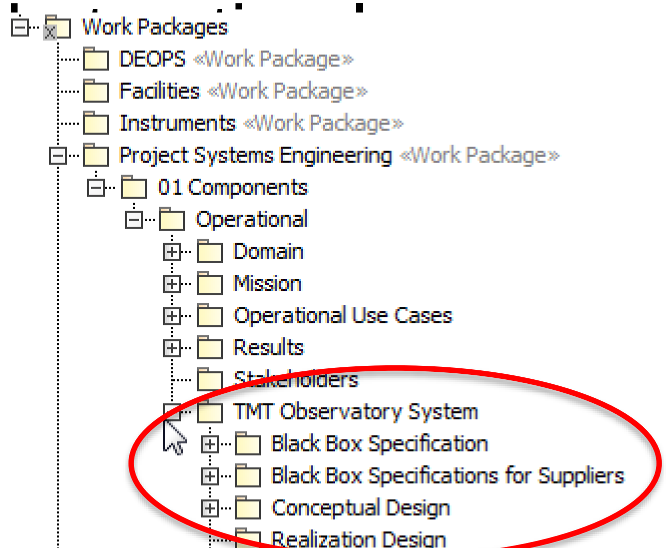
- Use MBSE to define **executable SysML model** that captures requirements, operational scenarios, behavior, system decomposition, relationships and between subsystems, etc.
- Use the model to **analyze the system design** for
  - Power consumption, mass, and duration/timing
  - Error budgets
- Produce **engineering documents**
  - Requirement Flow Down Document
  - Operational Scenario Document
  - Design Description Document
  - Interface Control Documents
- Uses **standard languages and techniques where practical to** avoid custom software development

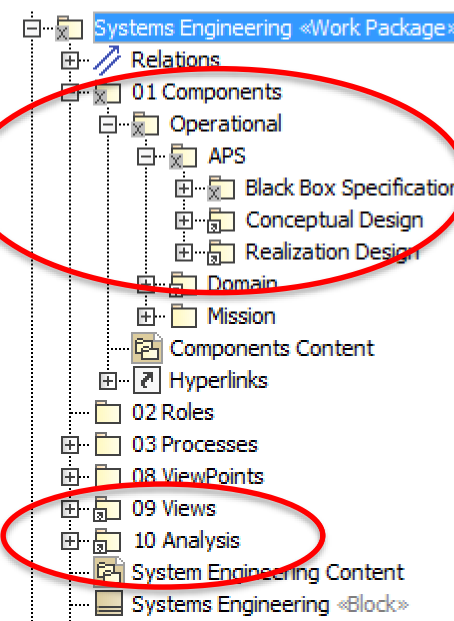
Model Walkthrough

Model Organization & Package Structure

- Organizing principles
  - Customer/Supplier relationship
  - Work Breakdown Structure







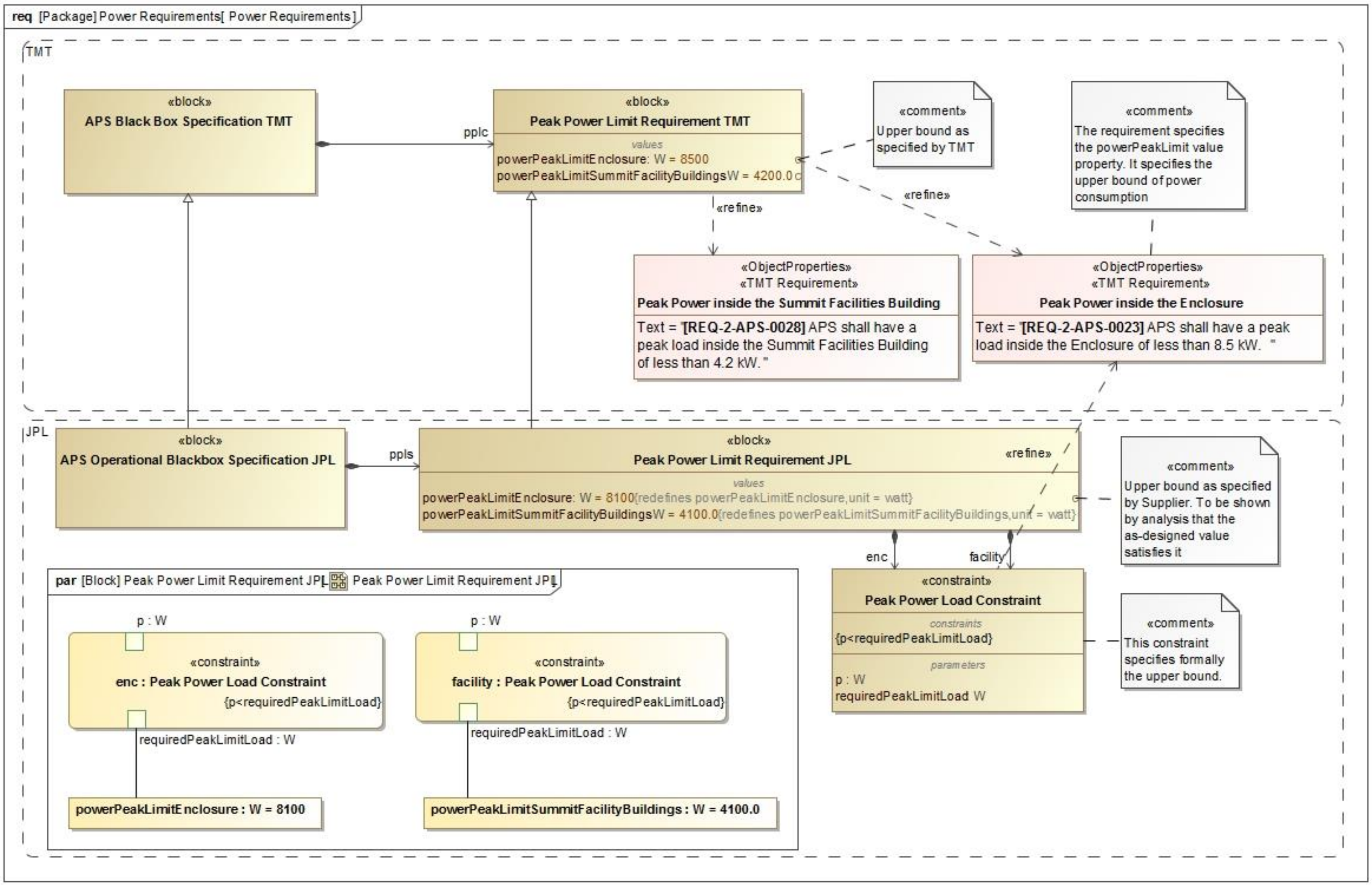
Executable System Engineering Method (ESEM)

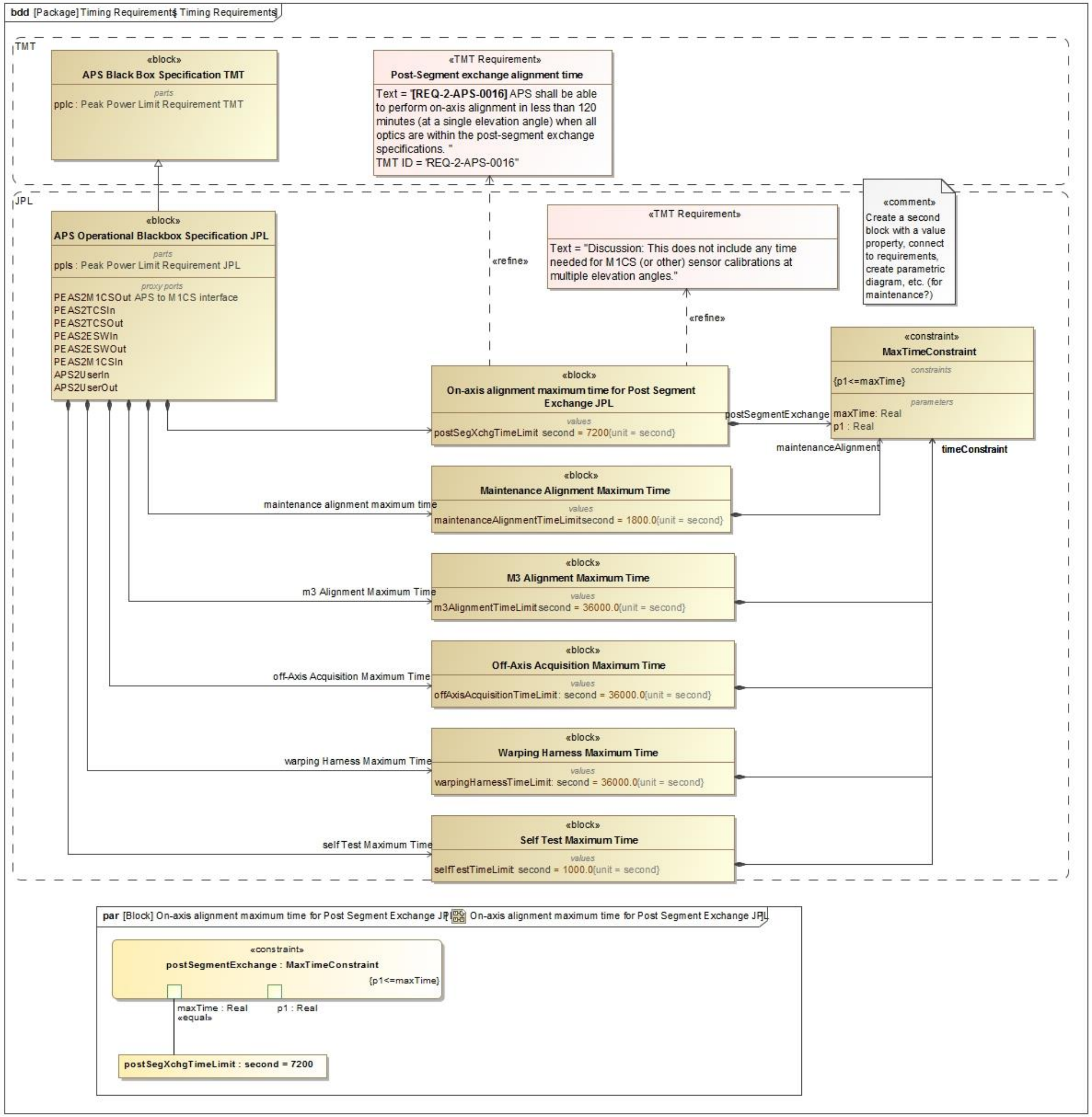
- Step 1: Formalize Requirements
- Step 2: Specify Design
- Step 3: Characterize Components
- Step 4: Specify Analysis Context
- Step 5: Specify Operational Scenarios
- Step 6: Specify Analysis Configurations
- Step 7: Run Analysis

Step 1: Formalize Requirements

- Requirement Pattern
  - Customer Side
    - Define the textual requirement with a Requirement
    - Optionally define a design black box specification with a Block with relevant value properties
    - Optionally refine the Requirement with a Constraint Block on the black box design Block
  - Supplier Side
    - Define a design black box specification with a Block (that refines the customer’s black box Block if any and provides tighter property values)
    - Refine the textual Requirement by a Constraint Block (if not already defined by the customer)

Step 1: Formalize Requirements



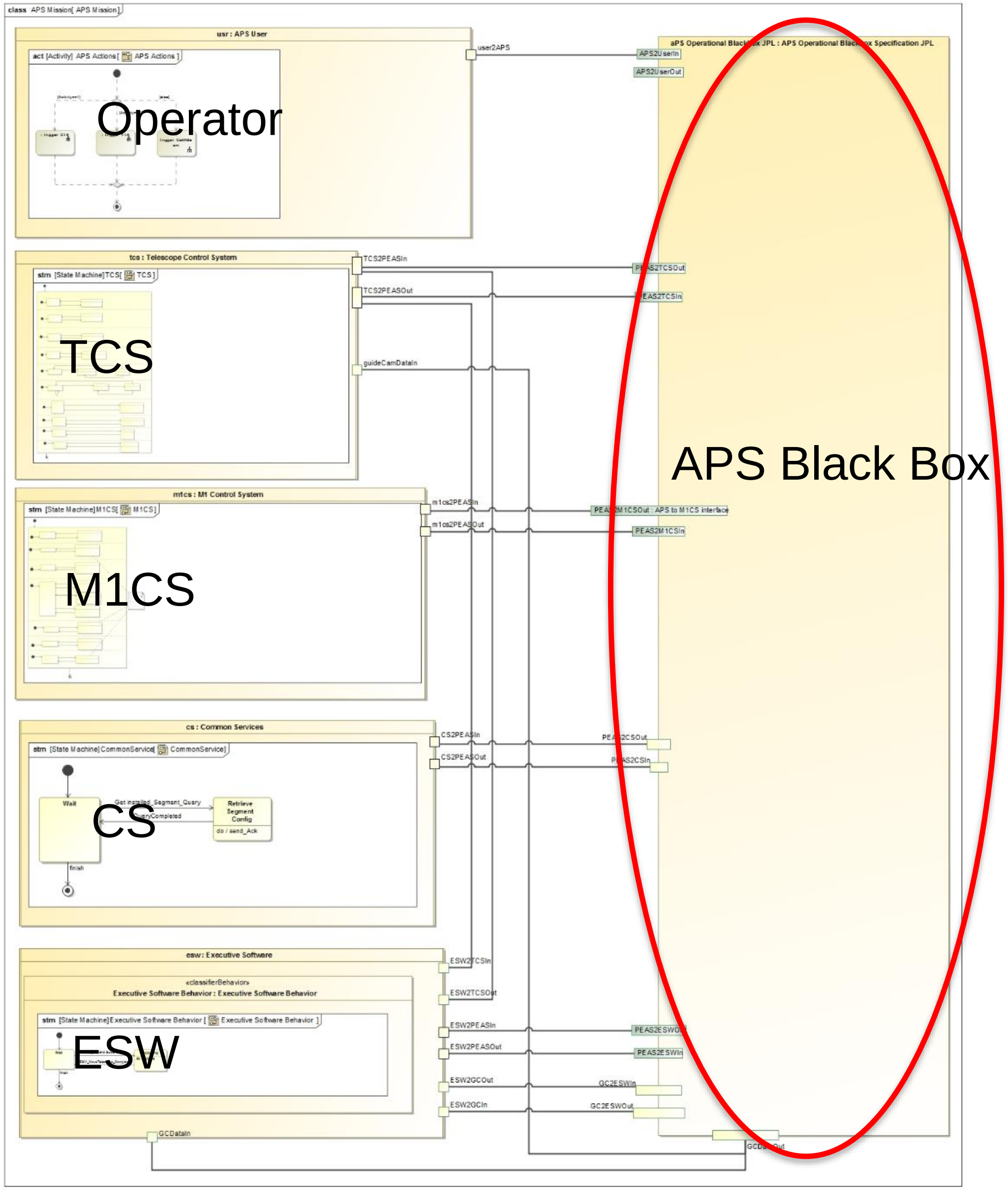

bdd [Package] Timing Requirements[Timing Requirements]
TMT
«block»
APS Black Box Specification TMT
parts
pplc : Peak Power Limit Requirement TMT
«TMT Requirement»
Post-Segment exchange alignment time
Text = "[REQ-2-APS-0016] APS shall be able to perform on-axis alignment in less than 120 minutes (at a single elevation angle) when all optics are within the post-segment exchange specifications. "
TMT ID = "REQ-2-APS-0016"
JPL
«block»
APS Operational Blackbox Specification JPL
parts
ppls : Peak Power Limit Requirement JPL
proxy ports
PEAS2M1CSOut APS to M1CS interface
PEAS2TCSIn
PEAS2TCSOut
PEAS2ESWIn
PEAS2ESWOut
PEAS2M1CSIn
APS2UserIn
APS2UserOut
«refine»
«TMT Requirement»
Text = "Discussion: This does not include any time needed for M1CS (or other) sensor calibrations at multiple elevation angles."
«refine»
«comment»
Create a second block with a value property, connect to requirements, create parametric diagram, etc. (for maintenance?)
«constraint»
MaxTimeConstraint
constraints
{p1<=maxTime}
parameters
maxTime: Real
p1 : Real
«block»
On-axis alignment maximum time for Post Segment Exchange JPL
values
postSegXchgTimeLimit second = 7200{unit = second}
postSegmentExchange
maintenanceAlignment
timeConstraint
«block»
Maintenance Alignment Maximum Time
values
maintenanceAlignmentTimeLimitsecond = 1800.0{unit = second}
maintenance alignment maximum time
«block»
M3 Alignment Maximum Time
values
m3AlignmentTimeLimit second = 36000.0{unit = second}
m3 Alignment Maximum Time
«block»
Off-Axis Acquisition Maximum Time
values
offAxisAcquisitionTimeLimit : second = 36000.0{unit = second}
off-Axis Acquisition Maximum Time
«block»
Warping Harness Maximum Time
values
warpingHarnessTimeLimit: second = 36000.0{unit = second}
warping Harness Maximum Time
«block»
Self Test Maximum Time
values
selfTestTimeLimit second = 1000.0{unit = second}
self Test Maximum Time
par [Block] On-axis alignment maximum time for Post Segment Exchange JPL[On-axis alignment maximum time for Post Segment Exchange JPL]
«constraint»
postSegmentExchange : MaxTimeConstraint
{p1<=maxTime}
maxTime : Real
«equal»
p1 : Real
postSegXchgTimeLimit : second = 7200

Step 2: Specify Design

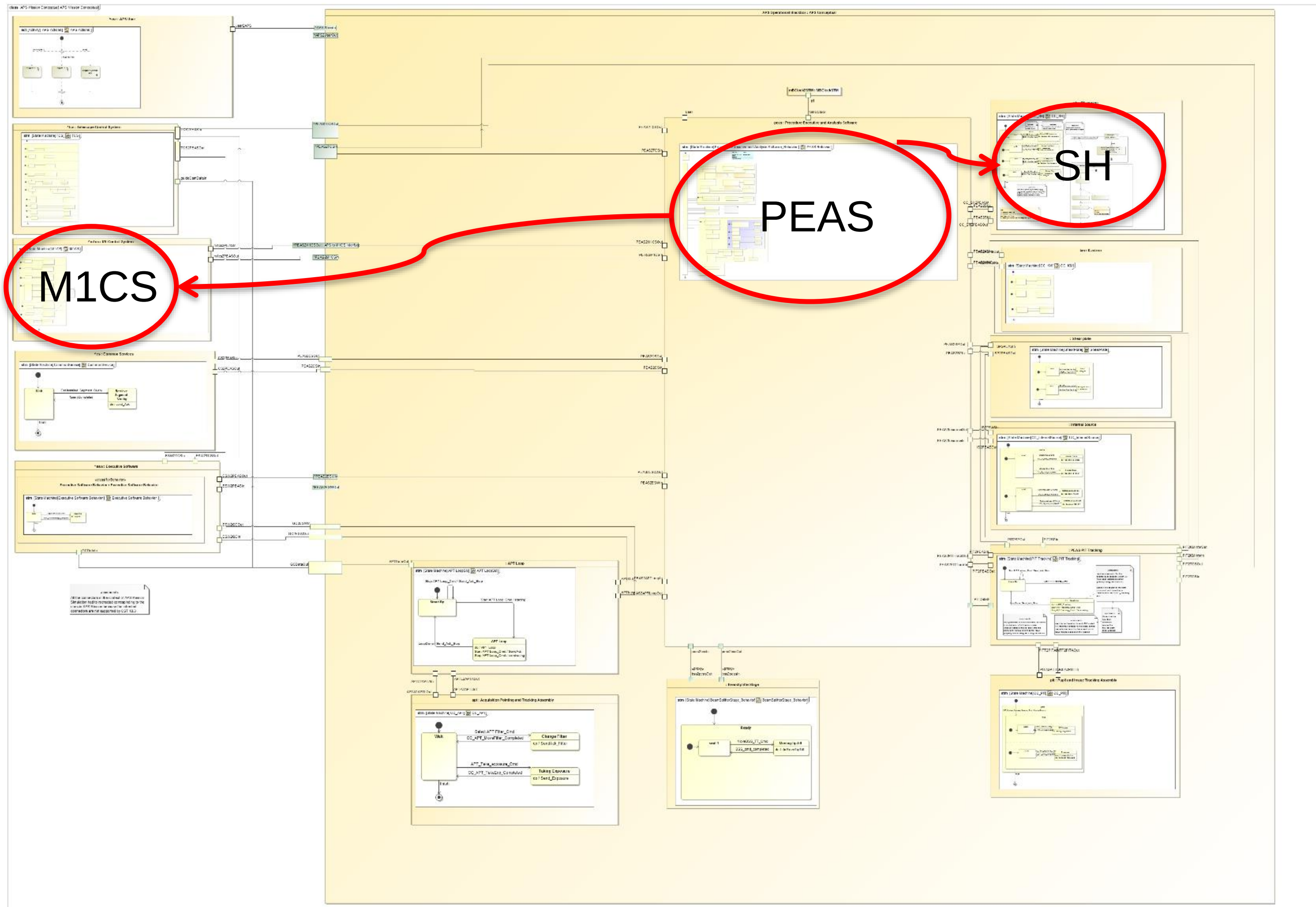
- Follow OOSEM to define two white box designs which specialize the black box design
  - Conceptual Specification
  - Realization Specification
- Decompose the white box designs into Blocks representing the subsystems

Black Box Design Model

Project level components communicate with APS black box block

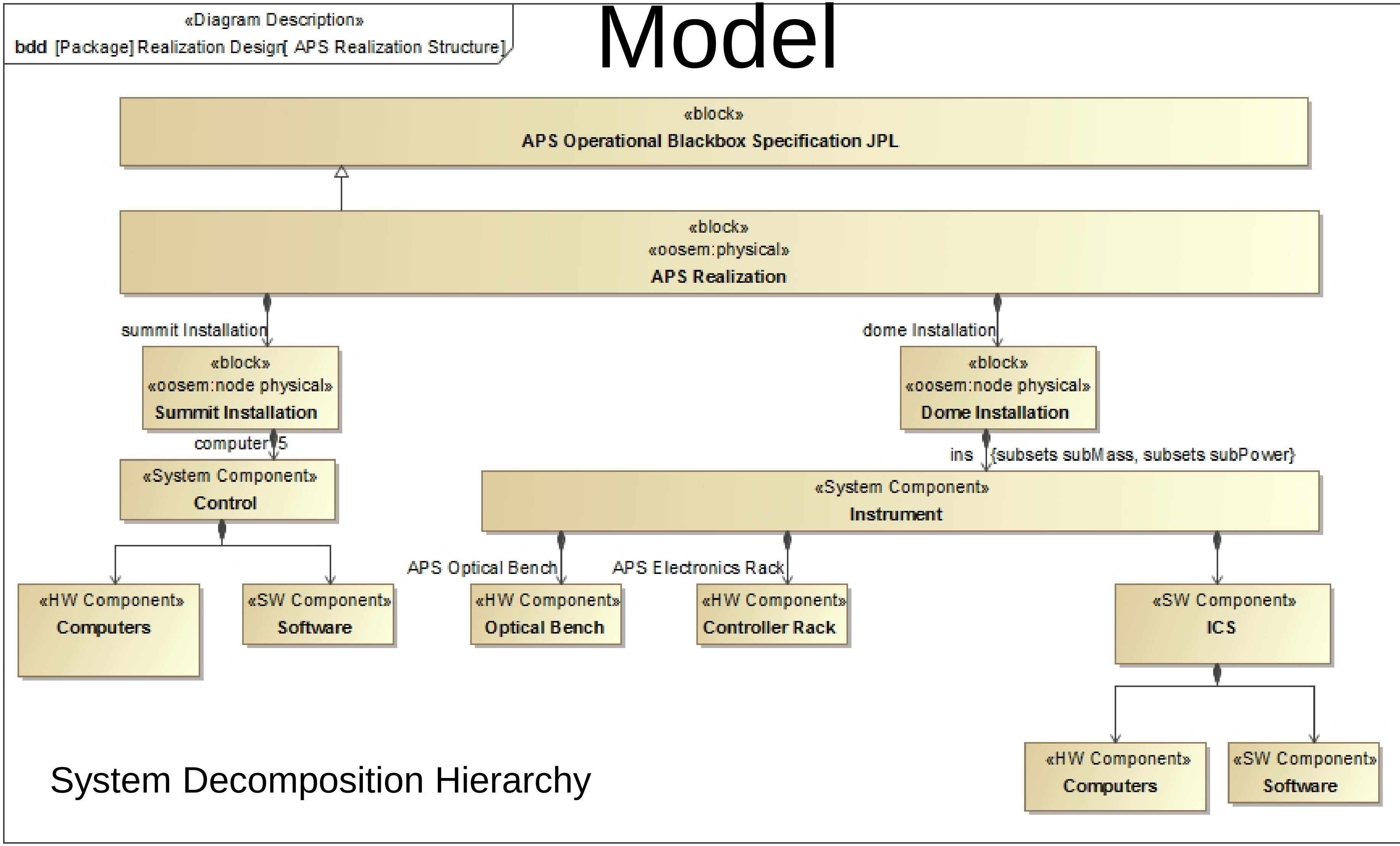


Step 2: Conceptual Design Model



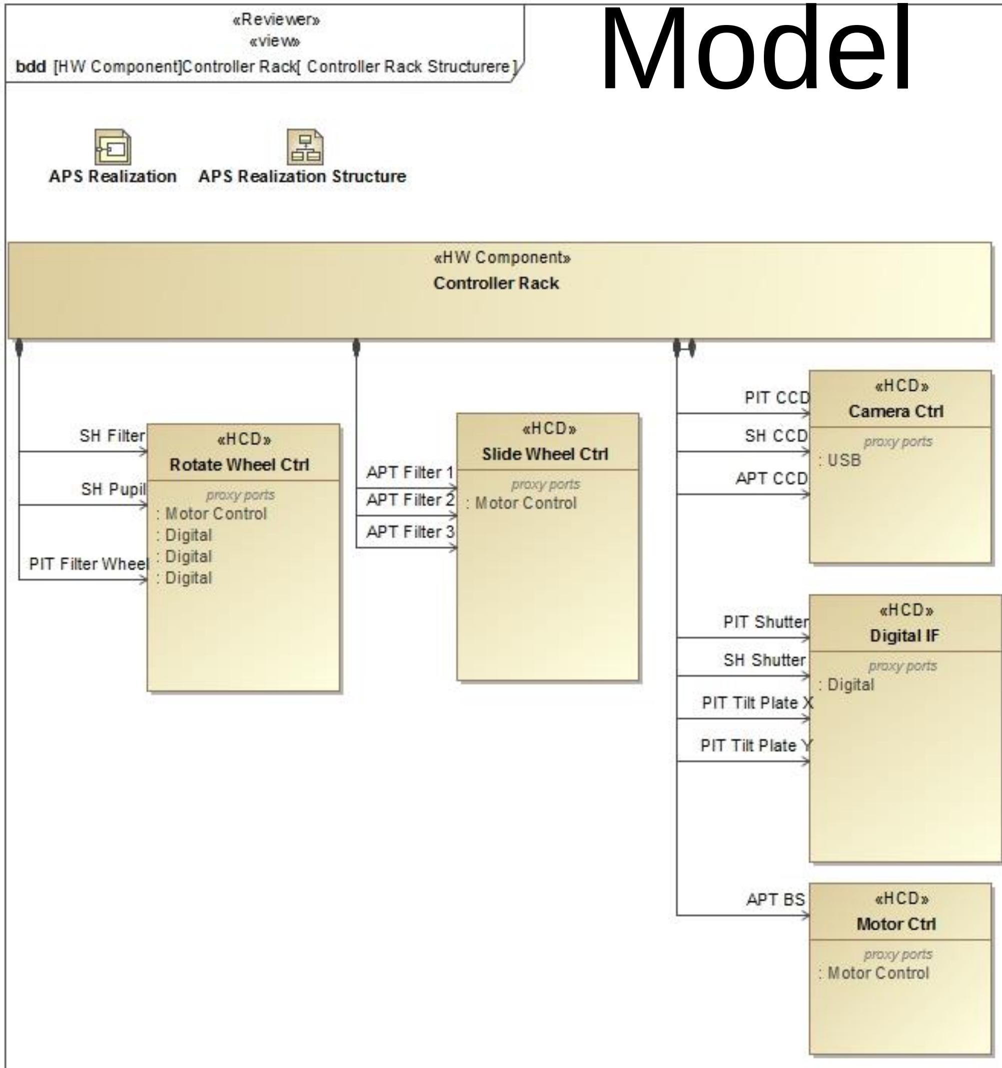
Communication between state machine specified components over ports

Step 2: Realization Design Model



System Decomposition Hierarchy

Step 2: Realization Design Model

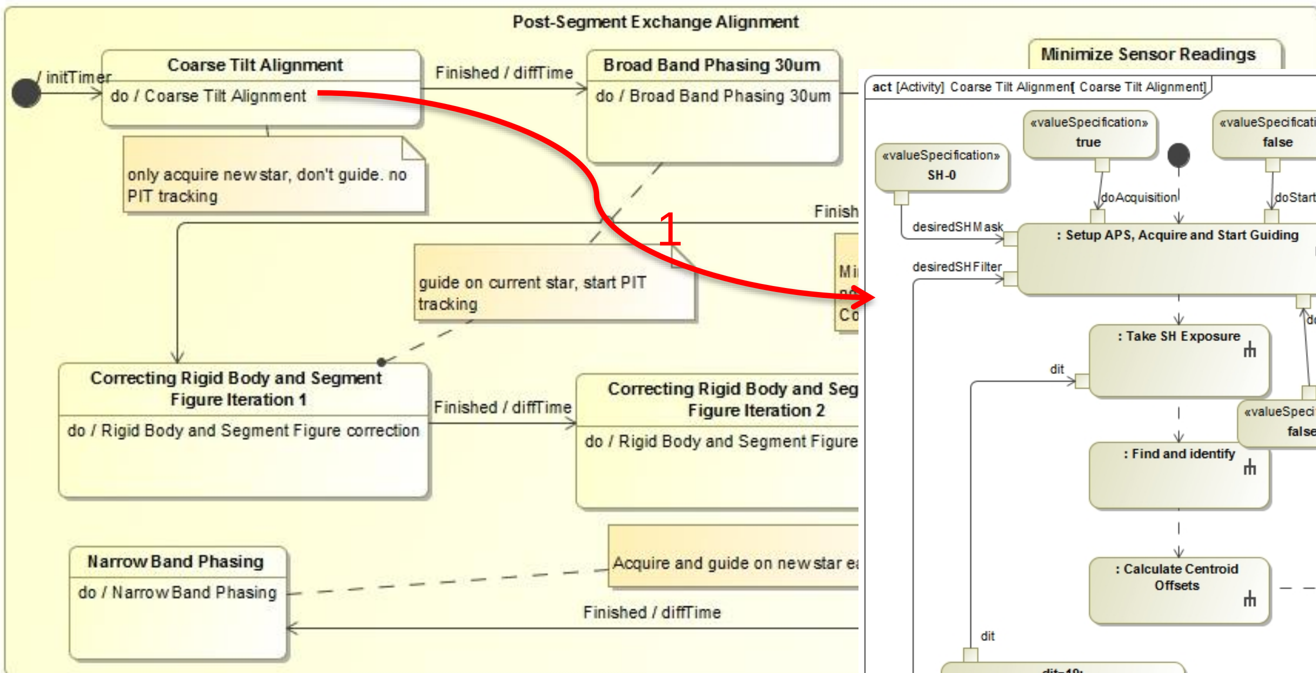


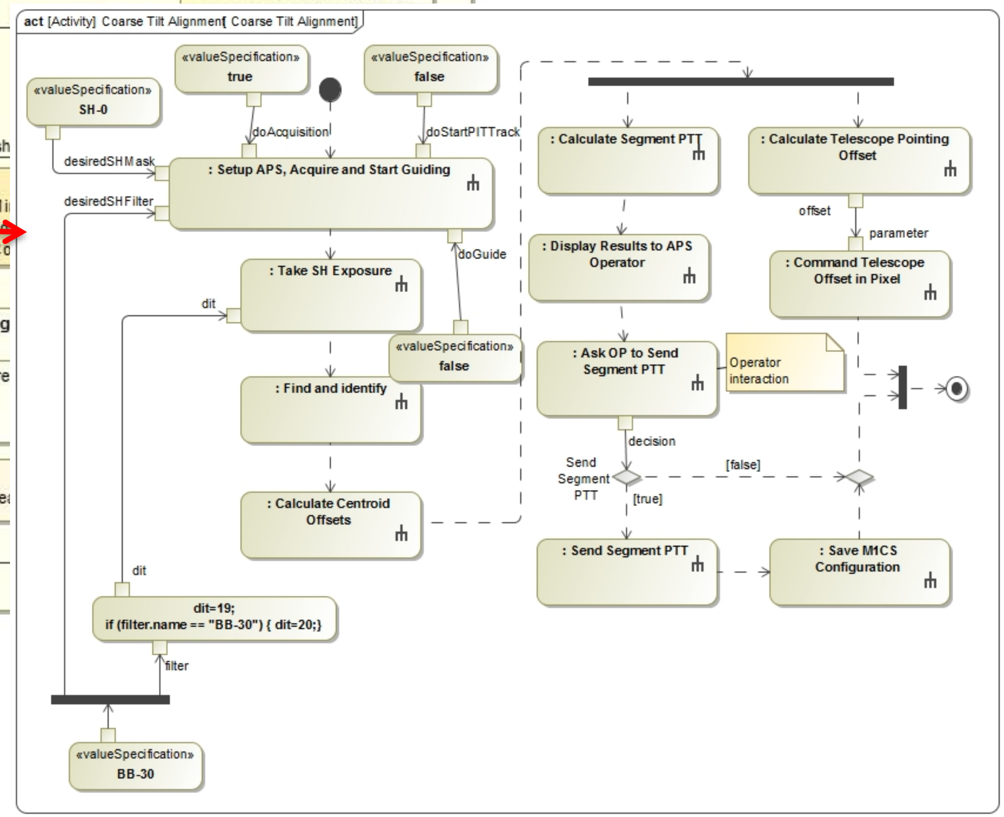
System Decomposition Hierarchy

Step 3: Characterize Components & Specify Behavior

- Specify behavior of components
  - E.g., using SysML state machines for lifecycle behavior
  - E.g., using SysML activity diagrams for functional flow
- Characterize Components, e.g., Using Patterns
- Example: Roll-up Pattern
  - Constrained value represents an aggregate value that is propagating up a hierarchy of subcomponents

Step 3: Characterize Conceptual Components



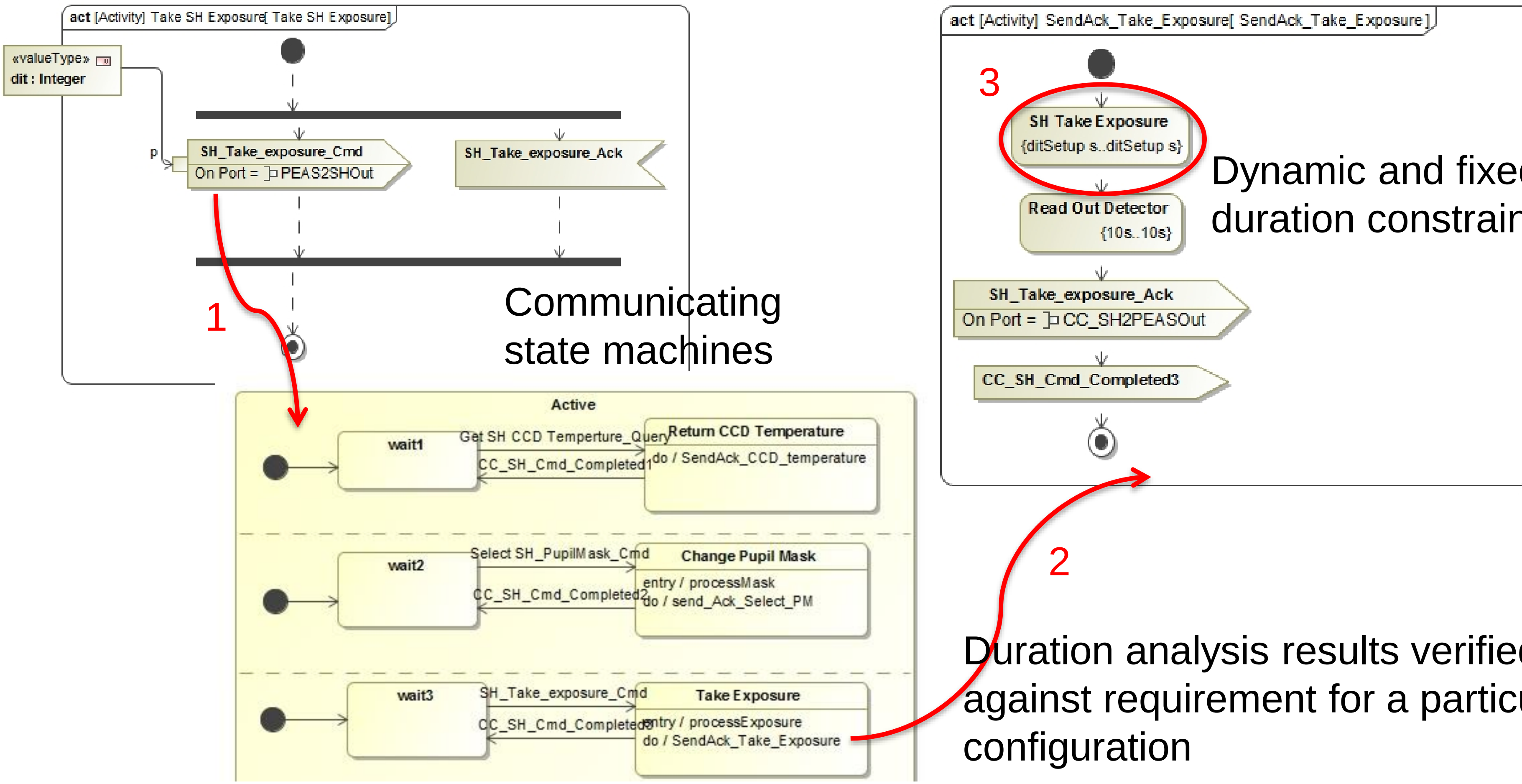


Operational behavior captured with state machines and activity models

2

#	Name	Specification	Constrained Element	Owner Of Constrained Element
1	dAPTMoving	10s..10s	Moving	SendAck_Filter
2	dAPTTakeExposure	ditSetup s..ditSetup s	APT Take Exposure	Send_Exposure
3	dCalcCentroidOffsets	0s..1s	Execute Centroid Offset Calculation	Calculate Centroid Offsets
4	dCalcPupilRegistrationOffset	0.5s..1s	Calculate Pupil Registration Offset	Calculate Pupil Registration and Image Offset
5	dCalcRMSForM2AndSegmentPTT	1s..3s	Calculate RMS for M2 and Segment PTT Cmds	Calculate RMS for M2 and Segment PTT cmds

Step 3: Characterize Conceptual Components



Communicating state machines

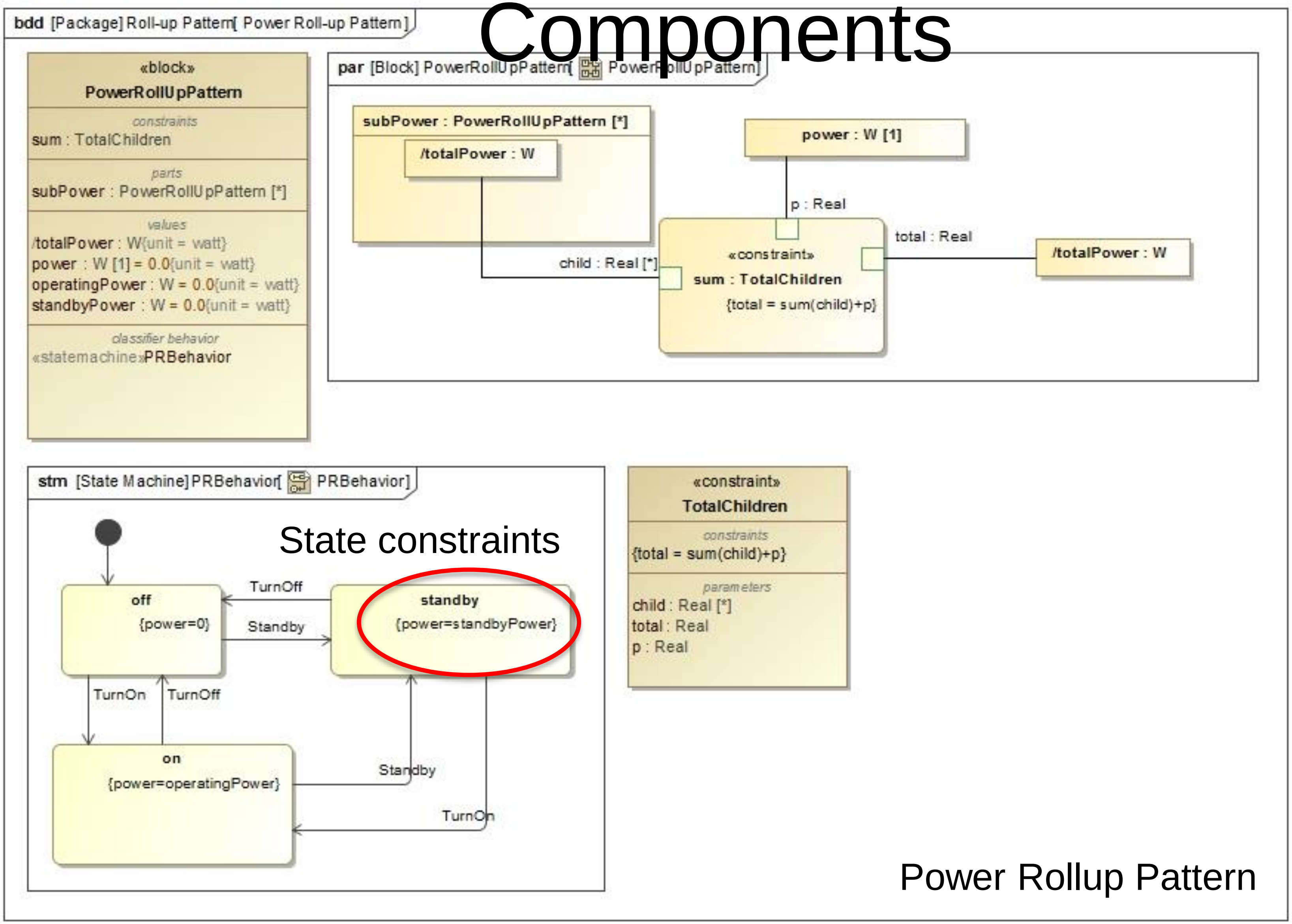
Dynamic and fixed duration constraints

Duration analysis results verified against requirement for a particular configuration

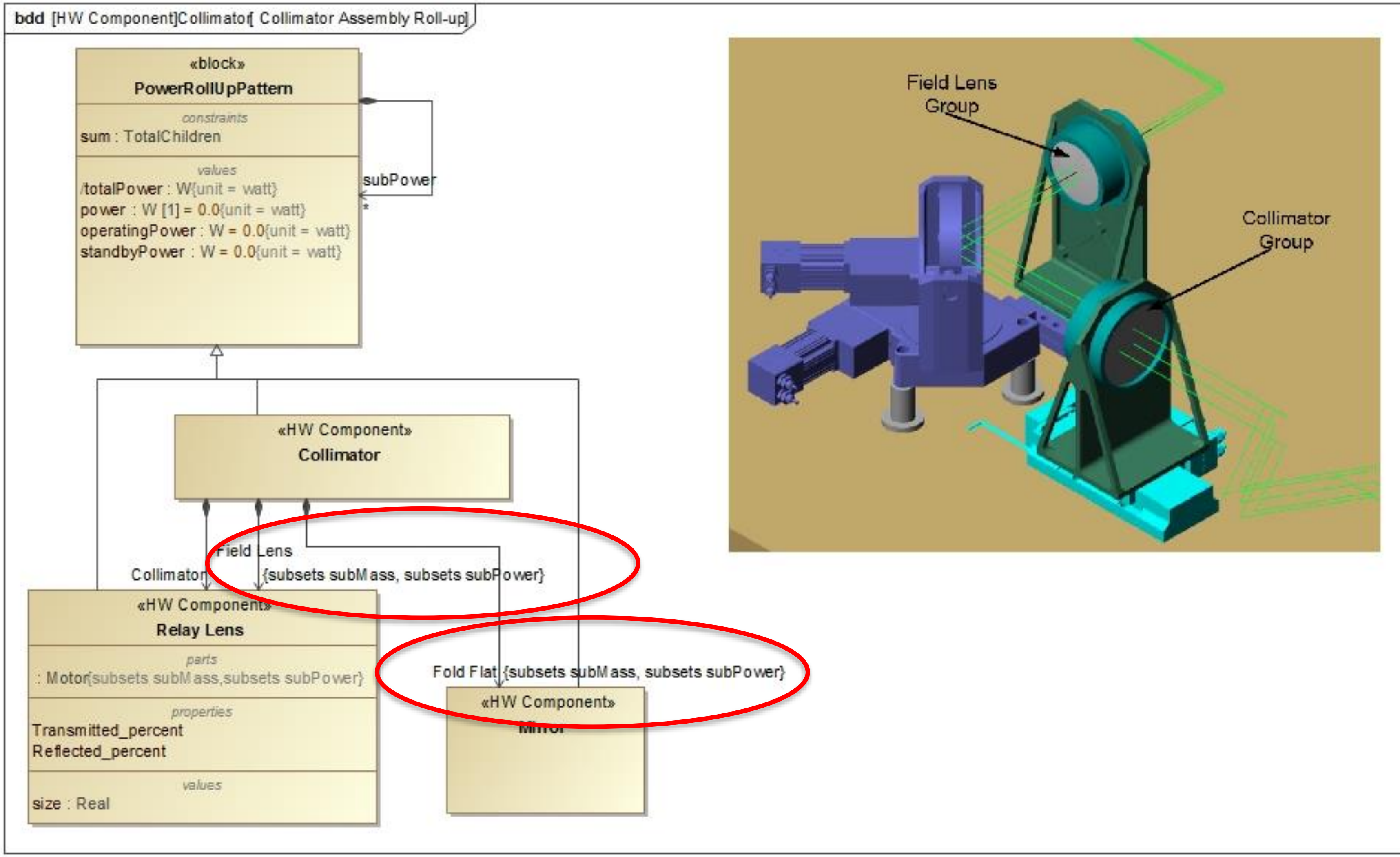
4

#	Name	T Final : Real	Post Seq Xchg Time Limit : Second	Post Segment Exchange : Post Segment Exchange Time Constraint	Off Axis Measurement Steps : Integer	Off Axis Map Points : Integer	RB Dit : Integer	Phasing Dit : Integer
1	maintenance Alignment Duration Scenario.aps mission conceptual.aps co		7200.0	pass				
2	maintenance Alignment Duration Scenario.aps mission conceptual.aps co	1517.0			6	7	45	20

Step 3: Characterize Realization Components



Step 3: Characterize Realization

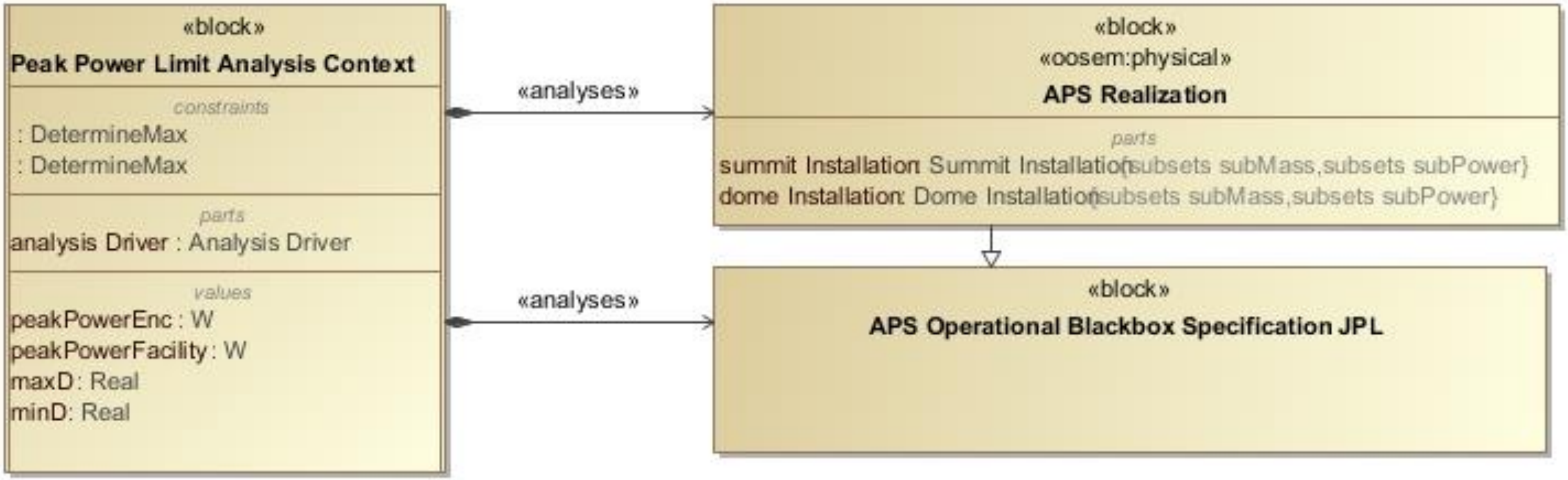


Power Roll-up Pattern Application

Step 4: Specify Analysis Context

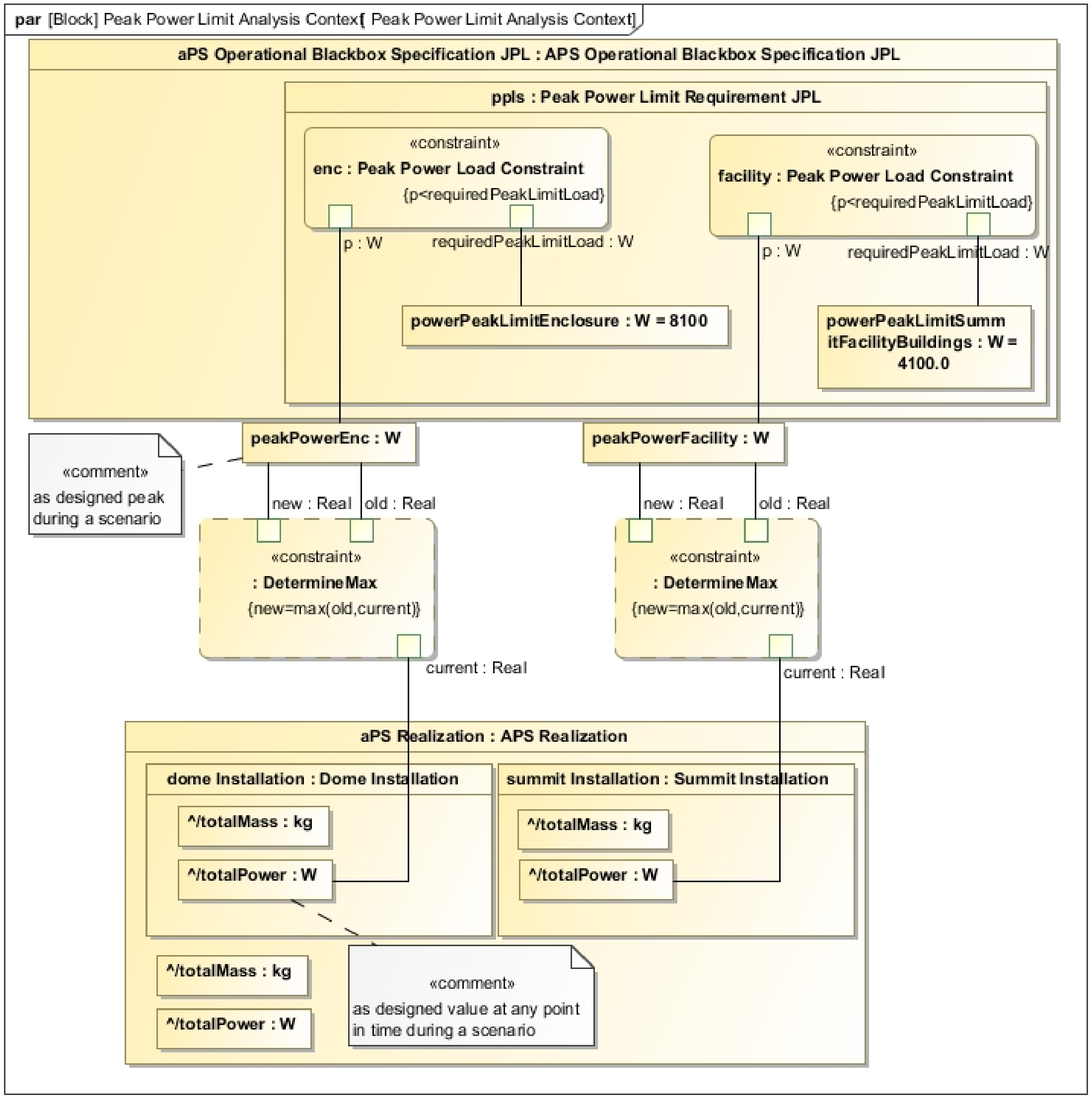
- Analysis Context Pattern
  - Abstract analysis context Block composes both the design black box Block and white box Block
  - Analysis properties defined on the analysis context Block (e.g., peak power, power margin)
  - Analysis parametric model on the analysis context that computes and binds analysis values

Step 4: Specify Analysis Context



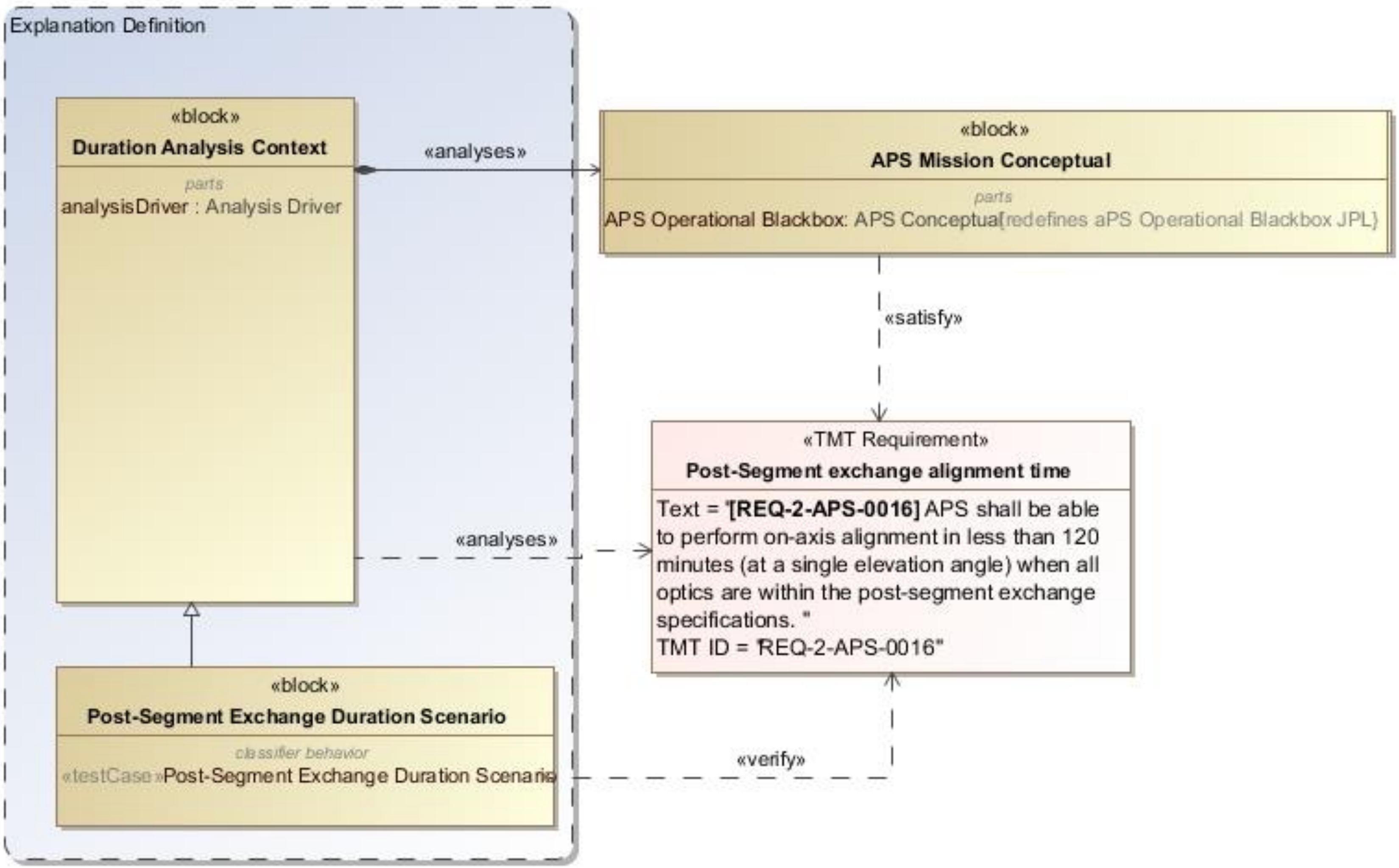
Analysis Context Pattern

Step 4: Specify Analysis Context

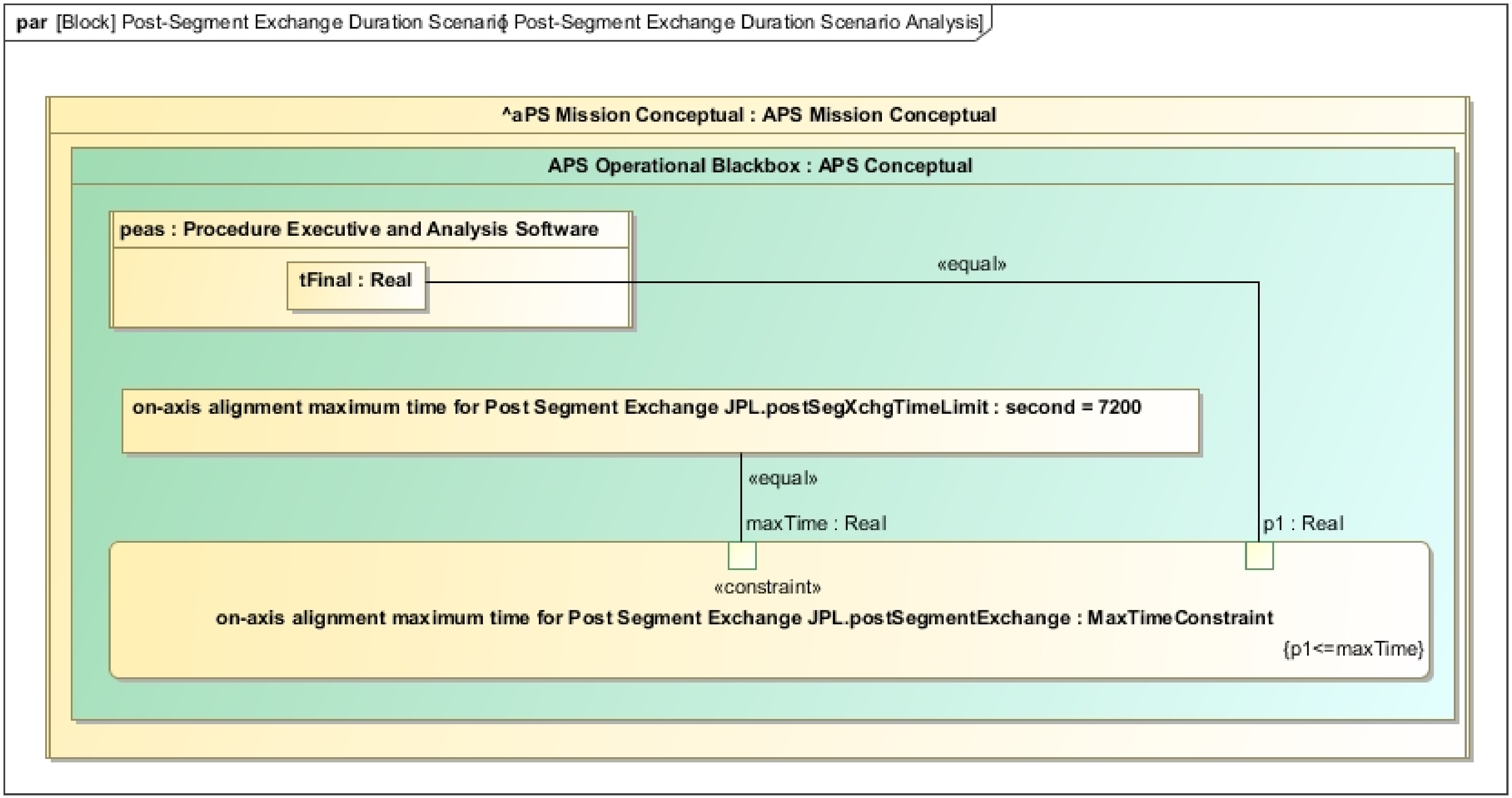


Analysis Context Parametric Model

Step 4: Specify Analysis Context



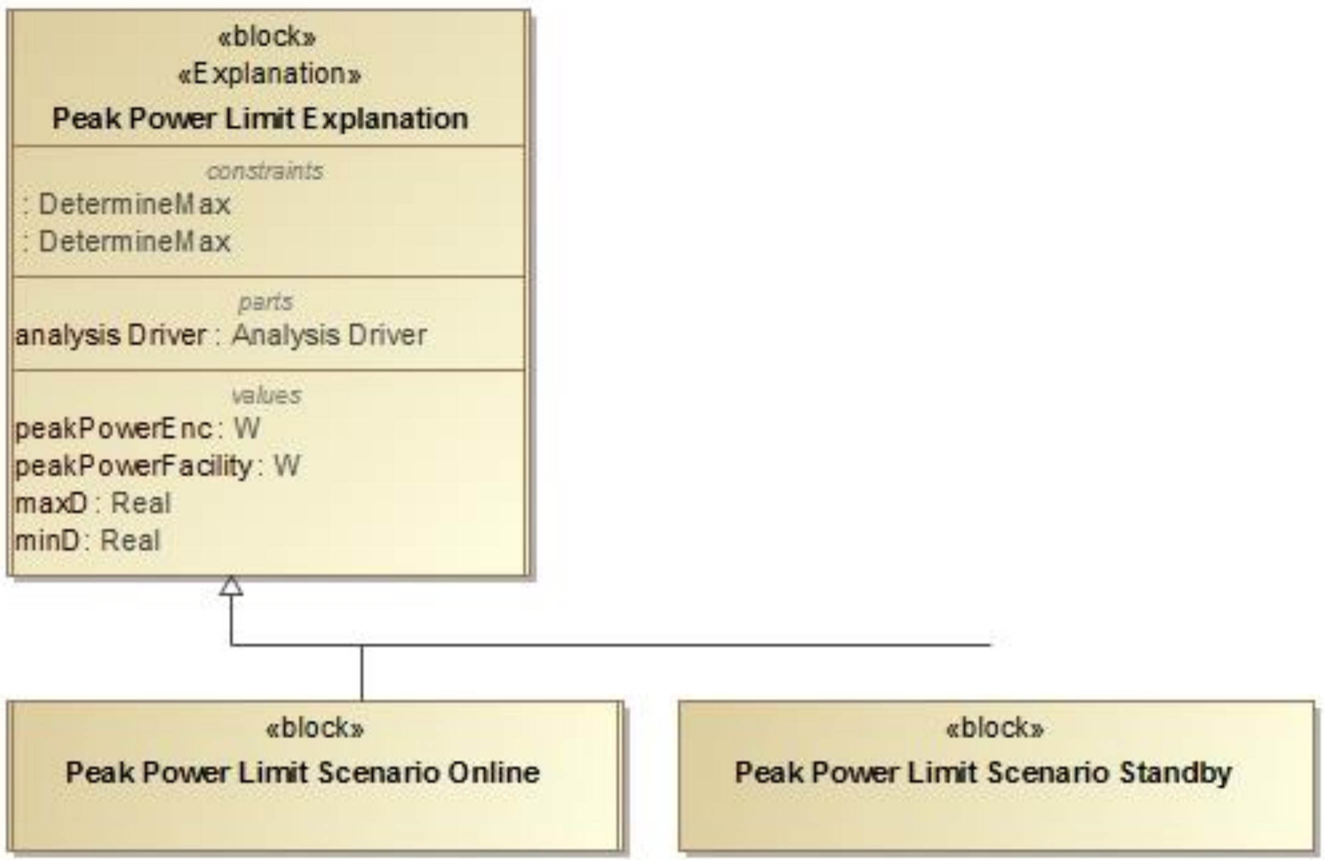
Step 4: Specify Analysis Context



Step 5: Specify Operational Scenarios

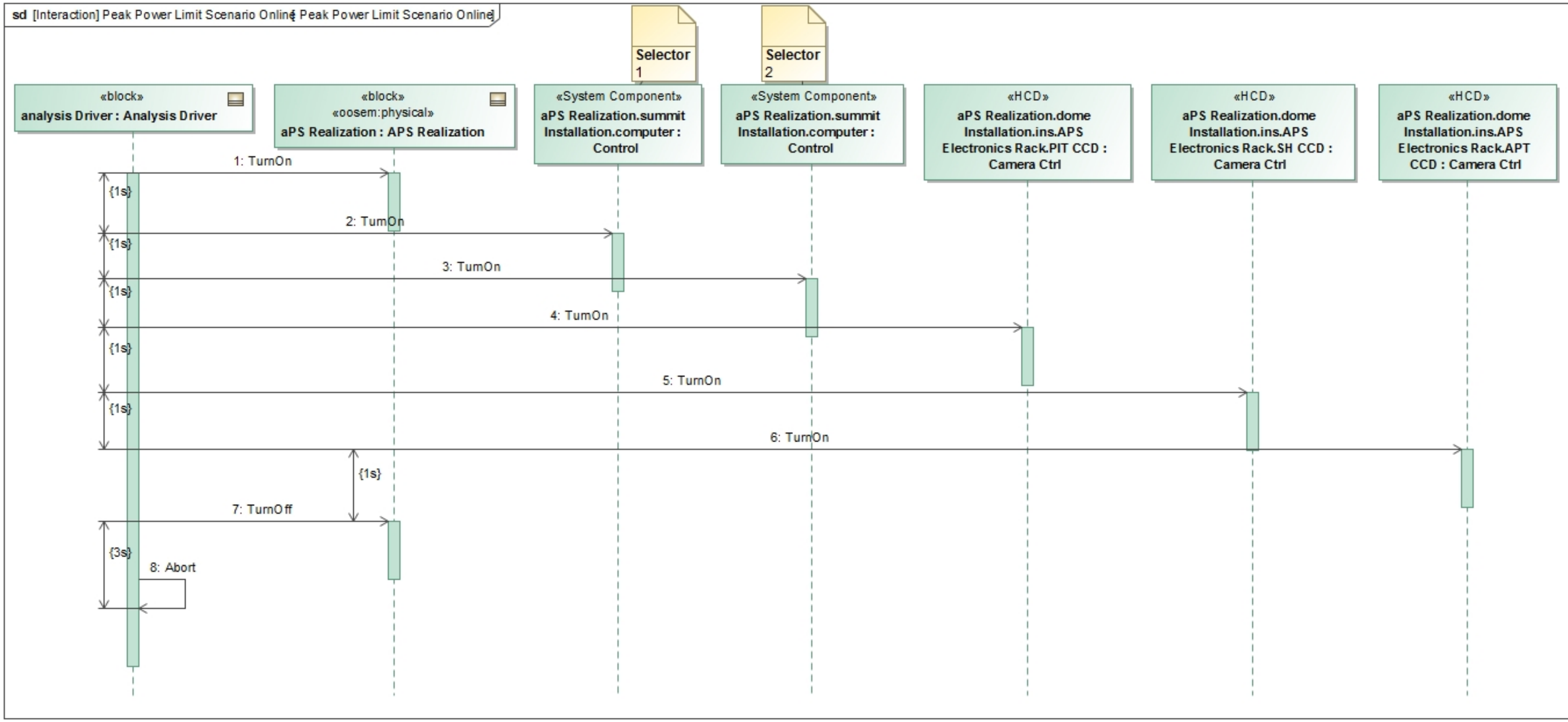
- Operational Scenario Pattern
  - Concrete analysis context Block which
    - Represents one operational scenario (e.g., power configuration)
    - Specializes the abstract analysis context Block
    - Redefines context’s properties with scenario-specific values
    - Defines an owned behavior (sequence diagram) as scenario driver
      - » Changes the states of the different components, by sending them signals, causing the rolling-up to occur automatically
      - » Can specify duration constraints to time the injection of signals thus controlling time spent in a certain state
      - » Can use state constraints (on components) to verify during execution if a component is actually in expected

Step 5: Specify Operational Scenarios



Operational Scenario Pattern

Step 5: Specify Operational Scenarios



Operational Scenario Driver

Step 6: Specify Scenario Configurations

- Scenario Condition Pattern
  - A decomposition tree of instance specifications representing the state of the scenario
    - Can be presented in tabular form
      - Rows represent the instance specifications (e.g., component)
      - Columns represent values (e.g., operating power) from the instance specifications
- Issues
  - Hard to keep instance specifications in sync with Block hierarchy
    - Mitigation: tool automation

Step 6: Specify Analysis Configurations

#	Name	Classifier	Operating Power : W	Standby Power : W
1	peak Power Limit Scenario Online.aPS Realization	APS Realization	0.0	0.0
2	peak Power Limit Scenario Online.aPS Realization.dome Installation	Dome Installation	0.0	0.0
3	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins	Instrument	0.0	0.0
4	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack	Controller Rack	0.0	0.0
5	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt bs	Motor Ctrl	0.0	0.0
6	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt bs.subMass[1]	MassRollUpPattern		
7	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt bs.subPower[1]	PowerRollUpPattern		
8	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt ccd	Camera Ctrl	150.0	200.0
9	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt ccd.subMass[1]	MassRollUpPattern		
10	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt ccd.subPower[1]	PowerRollUpPattern		
11	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt filter 1	Slide Wheel Ctrl	0.0	0.0
12	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt filter 1.subMass[1]	MassRollUpPattern		
13	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt filter 1.subPower[1]	PowerRollUpPattern		
14	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt filter 2	Slide Wheel Ctrl		
15	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.apt filter 3	Slide Wheel Ctrl		
16	peak Power Limit Scenario Online.aPS Realization.dome Installation.ins.aps electronics rack.pit ccd	Camera Ctrl	150.0	100.0

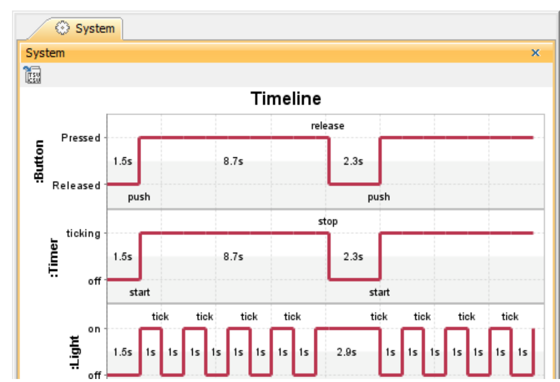
Scenario Initial Condition Pattern

177	peak Power Limit Scenario Online.aPS Realization.summit Installation.computer1.subPower[1]	PowerRollUpPattern				
178	peak Power Limit Scenario Online.aPS Realization.summit Installation.subMass[1]	MassRollUpPattern				
179	peak Power Limit Scenario Online.aPS Realization.summit Installation.subPower[1]	PowerRollUpPattern				
180	peak Power Limit Scenario Online.aPS Operational Blackbox Specification JPL.pplc	Peak Power Limit Requirem				8500.0
181	peak Power Limit Scenario Online.aPS Operational Blackbox Specification JPL.ppls	Peak Power Limit Requirem				8100.0
182	peak Power Limit Scenario Online.aPS Realization.pplc	Peak Power Limit Requirem				
183	peak Power Limit Scenario Online.aPS Realization.ppls	Peak Power Limit Requirem				
184	peak Power Limit Scenario Online.aPS Realization.summit Installation.computer1	Control	500.0	100.0	0.0	

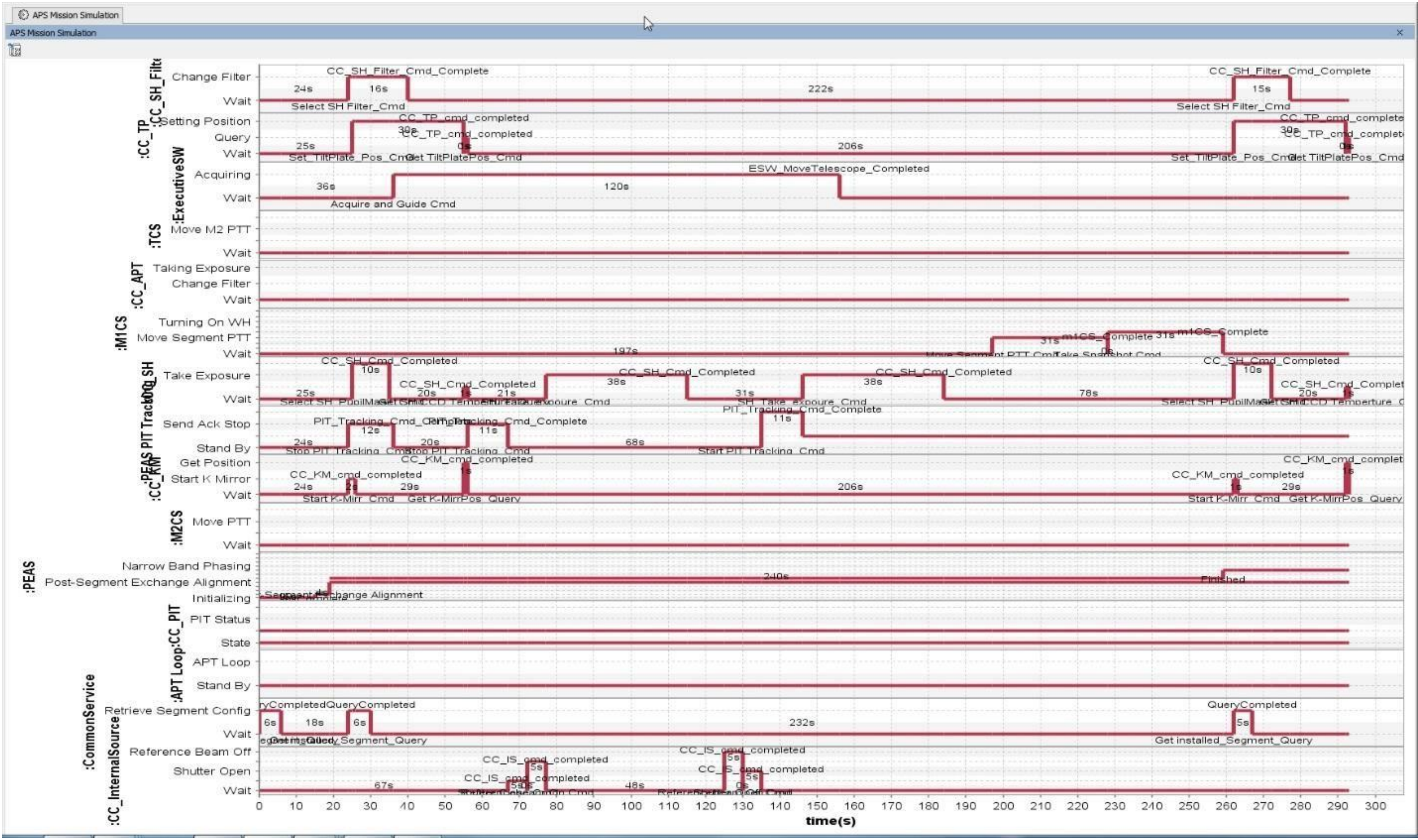
Step 7: Run Analysis

- Run the configured analysis with a simulation engine on the initial conditions to get the final conditions:
- Produce the following views on final conditions
  - **Table** showing final analysis values (e.g., peak power) and the constraint's pass/fail status for each scenario
  - **Timelines**: state changes for components over time
  - **Value profiles**: total rolled up values over time

#	Name	Classifier	T Final : Real	Ph
1	calibrations Duration S	Calibrations Duration S		
2	calibrations Duration S	APS Conceptual		
3	calibrations Duration S	Procedure Executive ar	8466.0	11



Timeline of component states



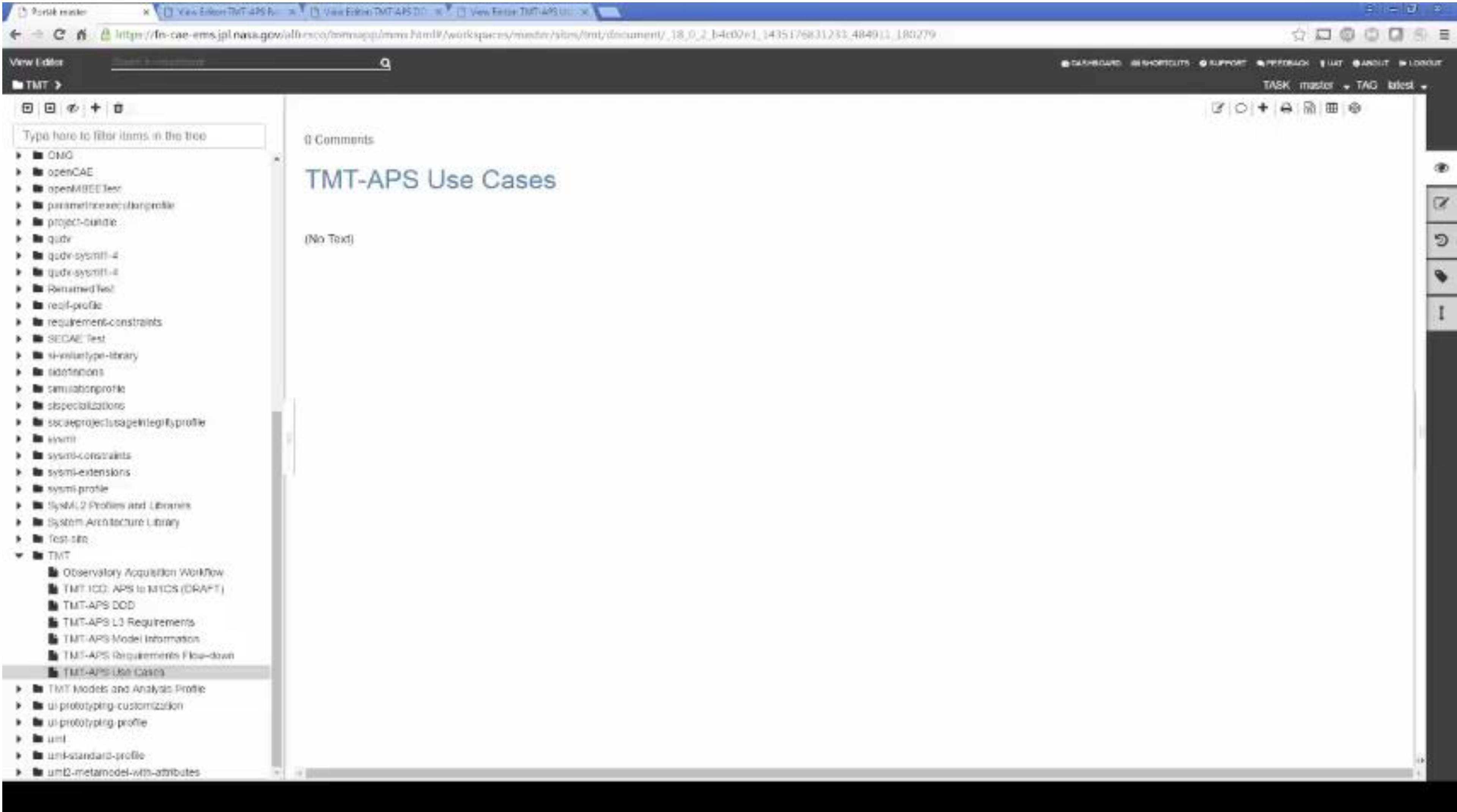
Duration Analysis results

#	Name	Classifier	Post Seg Xchg Time Limit : Second	T Final : Real	Post Segment Exchange : Max Time Constraint	Broadband Phasing Steps : Integer	Narrowband Filter Steps : Integer	Rigid Body Steps : Integer	RB Dit : Integer	Phasing Dit : Integer
1	post-Segment Exchange Du	Post-Segment Exchange Du								
2	post-Segment Exchange Du	Post-Segment Exchange Du								
3	post-Segment Exchange Du	Post-Segment Exchange Du								
4	post-Segment Exchange Du	Post-Segment Exchange Du								
5	post-Segment Exchange Du	Post-Segment Exchange Du								
6	post-Segment Exchange Du	APS Conceptual								
7	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
8	post-Segment Exchange Du	Procedure Executive and Ar		4605.0		11	2	6	45	20
9	post-Segment Exchange Du	APS Conceptual								
10	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
11	post-Segment Exchange Du	Procedure Executive and Ar		4577.0		11	2	6	45	20
12	post-Segment Exchange Du	APS Conceptual								
13	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
14	post-Segment Exchange Du	Procedure Executive and Ar		4551.0		11	2	6	45	20
15	post-Segment Exchange Du	APS Conceptual								
16	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
17	post-Segment Exchange Du	Procedure Executive and Ar		4516.0		11	2	6	45	20
18	post-Segment Exchange Du	APS Conceptual								
19	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
20	post-Segment Exchange Du	Procedure Executive and Ar		4517.0		11	2	6	45	20
21	post-Segment Exchange Du	APS Conceptual								
22	post-Segment Exchange Du	On-axis alignment maximum	7200.0		pass					
23	post-Segment Exchange Du	Procedure Executive and Ar		4913.0		11	2	6	45	20

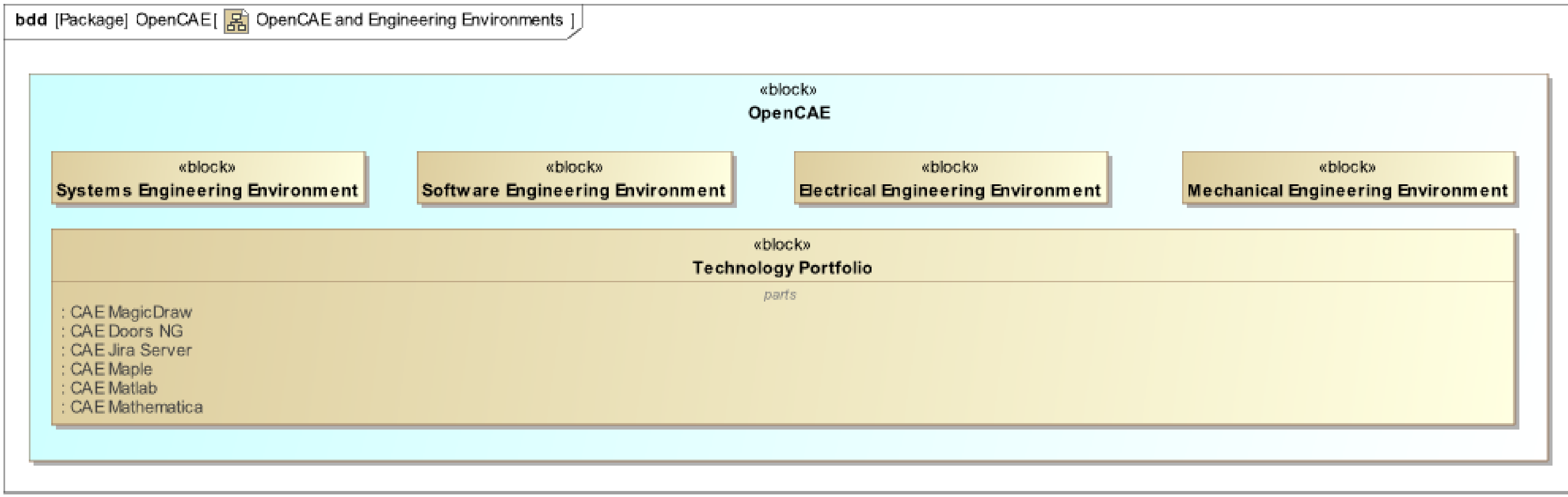
Power Analysis Results

#	Name	Classifier	Peak Power Enc : W	Peak Power Facility : W	Power Peak Limit Enclosure : W	Enc : Peak Power Load Constraint	Power Peak Limit Summit Facility Buildings : W	Facility : Peak Power Load Constraint
1	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
2	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
3	peak Power Limit Scenario Online.aps realization1.peak Power Lim	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
4	peak Power Limit Scenario Online.aps realization.peak Power Limit	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
5	peak Power Limit Scenario Online at 2016.04.30 00.49	Peak Power Limit Scenario Online	420.0	500.0				
6	peak Power Limit Scenario Online at 2016.04.30 00.52	Peak Power Limit Scenario Online	420.0	500.0				
7	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
8	peak Power Limit Scenario Online.aps realization2.peak Power Lim	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
9	peak Power Limit Scenario Online at 2016.04.30 00.54	Peak Power Limit Scenario Online	420.0	500.0				
10	peak Power Limit Scenario Online at 2016.07.25 14.28	Peak Power Limit Scenario Online	460.0	500.0				
11	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
12	peak Power Limit Scenario Online.aps realization3.peak Power Lim	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
13	peak Power Limit Scenario Online at 2016.09.29 15.38	Peak Power Limit Scenario Online	460.0	500.0				
14	peak Power Limit Scenario Online at 2017.02.26 18.29	Peak Power Limit Scenario Online	460.0	500.0				
15	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
16	peak Power Limit Scenario Online.aps operational blackbox specifi	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
17	peak Power Limit Scenario Online.aps realization4.peak Power Lim	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass
18	peak Power Limit Scenario Online.aps realization5.peak Power Lim	Peak Power Limit Requirement JPL			8100.0	pass	4100.0	pass

System Level Analysis



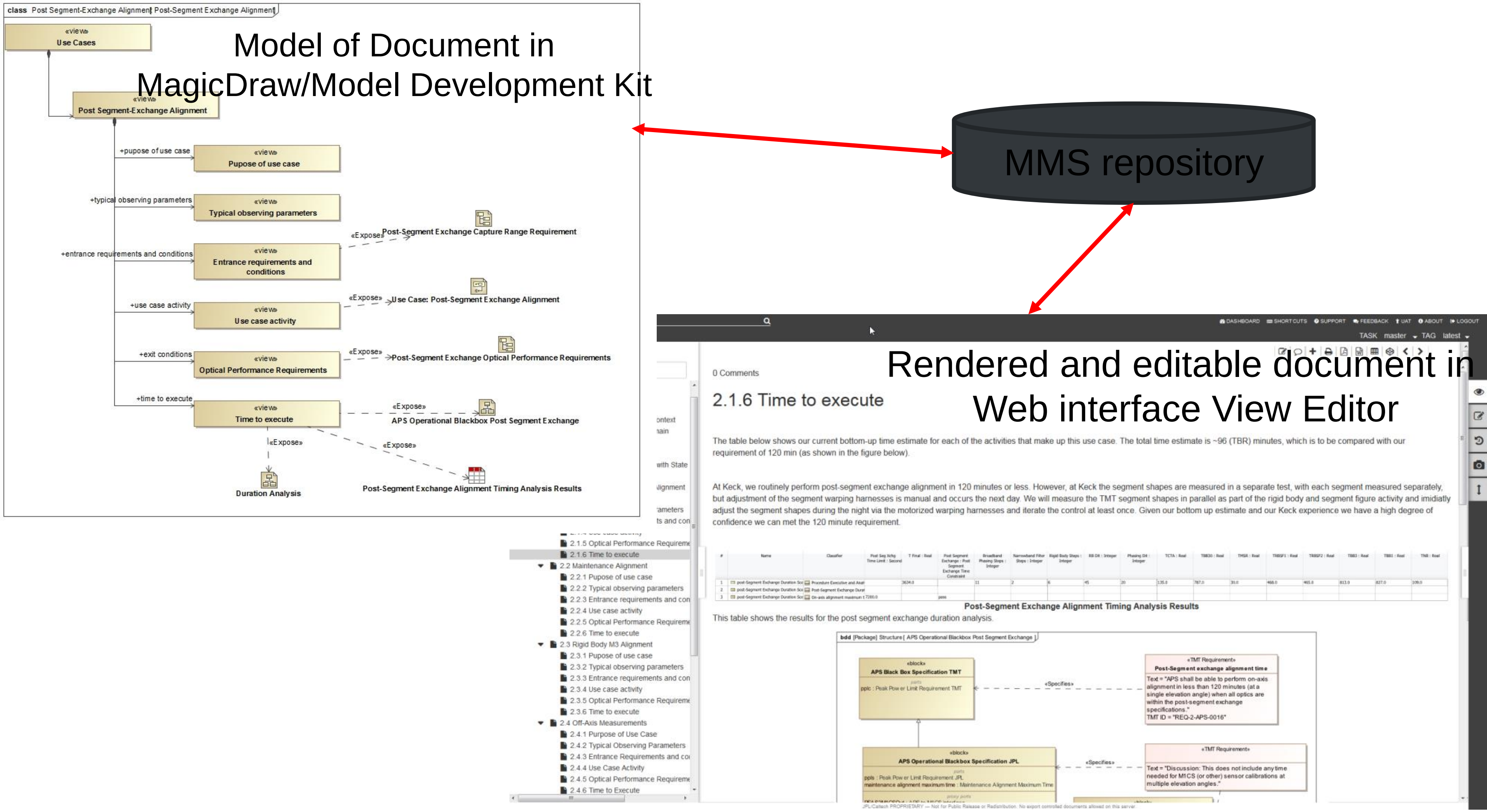
Environments



OpenMBEE
https://open-mbee.github.io/

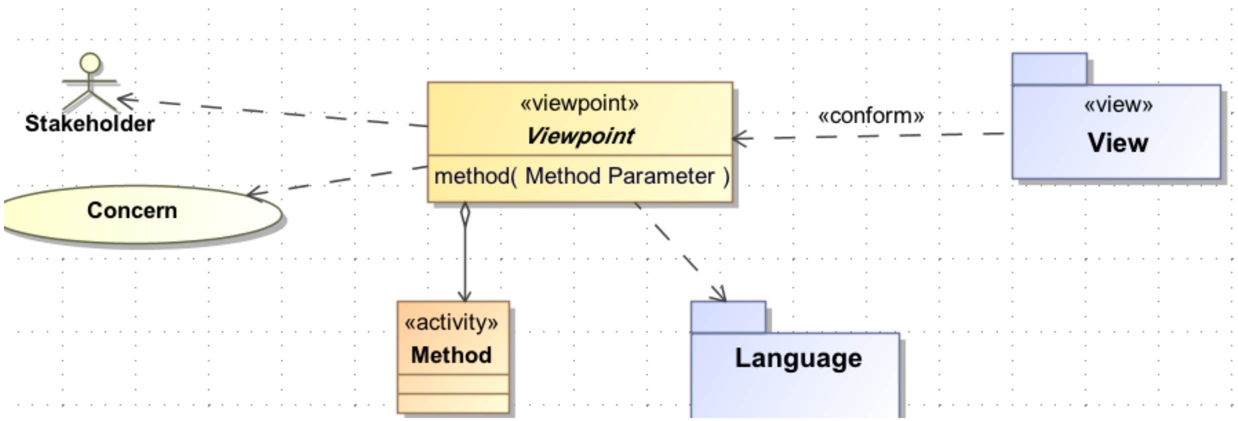
- OpenMBEE provides a platform for modeling that utilizes the Model Management System (MMS) that can be accessed from rich SysML desktop clients like MagicDraw, light-weight web-based client like View Editor, mathematical computation programs like Mathematica, and any other tool that can utilize RESTful web services.
- The model repository provides the following features:
  – Basic Infrastructure for Version, Workflow, Access Control
  – Flexibility of content
  – Support for Web Applications and Web-based API access
  – Multi-tool and multi-repository integration across engineering and management disciplines

OpenMBEE Core Integration



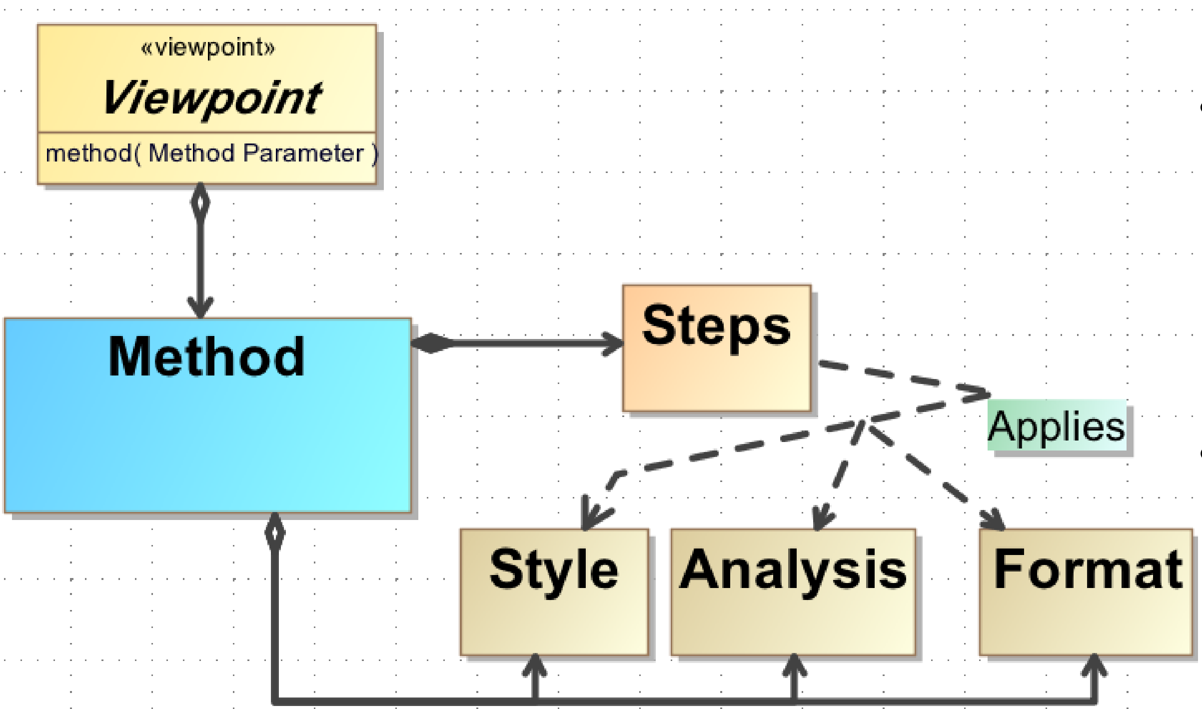
Building the Viewpoint Model

- Viewpoint Model
  - Purpose informed by Stakeholder Concerns
  - Methods and Analysis for constructing the View from the Model
  - Presentation Rules



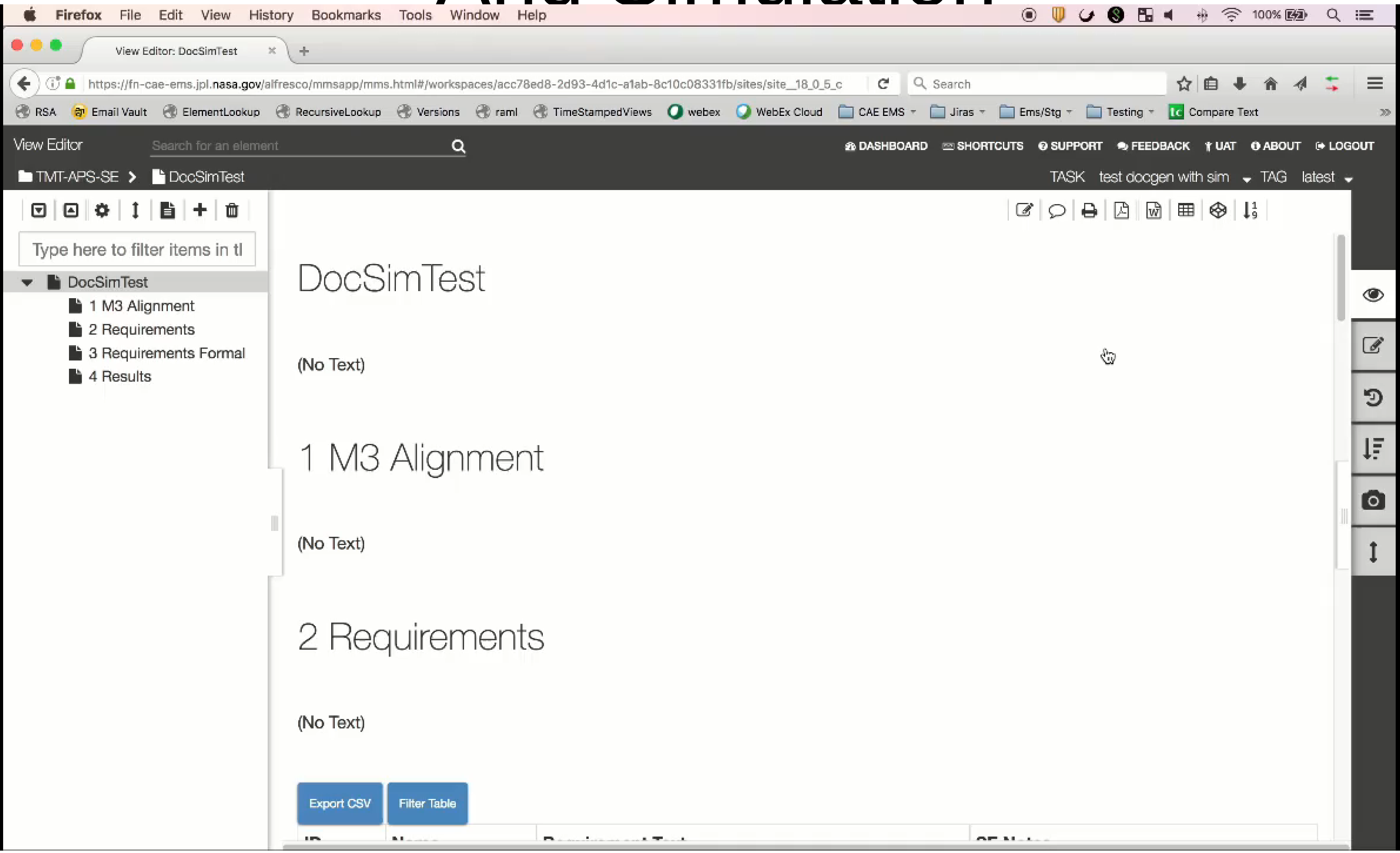
Docgen
https://github.com/Open-MBEE/mdk/tree/mdk-manual/src/main/dist/manual

Method and Analysis



- Methods
  - Ordered steps for producing the View
- Analysis
  - describe the nature of queries of the model
  - Analytical assertions
  - Rules for completeness and consistency
- Format and Presentation Style
  - Describe the conventions, styles and formats for how the information is presented in the View

Integrated Document Generation And Simulation



Summary

- Models are ubiquitous in domain engineering
- Still many disconnected document based artifacts
- Integration of requirements and behavioral/performance model
- Method and infrastructure exist to tie in system level models into domain specific models
- Leverage model executability (OOSEM + ESEM) and Co-simulation (FMI) and Co-Analysis
- Automated requirements verification of architecture and design
- Consistent Model Based Project Documentation

Acknowledgments

The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and NoMagic.

The TMT Project gratefully acknowledges the support of the TMT collaborating institutions. They are the Association of Canadian Universities for Research in Astronomy (ACURA), the California Institute of Technology, the University of California, the National Astronomical Observatory of Japan, the National Astronomical Observatories of China and their consortium partners, and the Department of Science and Technology of India and their supported institutes. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the National Research Council of Canada, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Universities for Research in Astronomy (AURA) and the U.S. National Science Foundation.

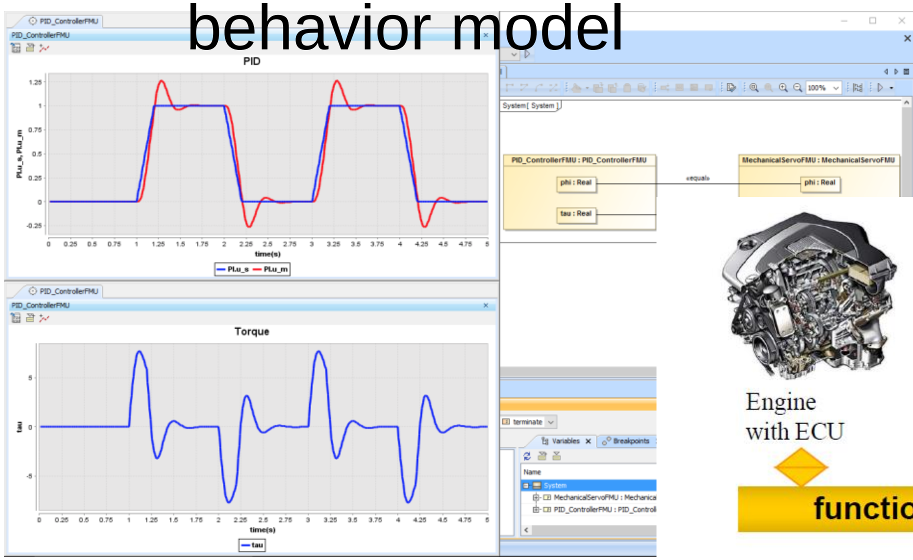
References

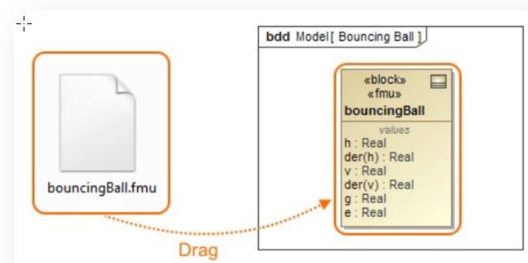
- Karban, R., Jankevičius, N., Elaasar, M. “ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns”, (to appear in the proceedings of INCOSE International Symposium (IS), Edinburgh, Scotland, 2016.)
- Karban R., Dekens F., Herzig S., Elaasar M, Jankevičius N., “Creating systems engineering products with executable models in a model-based engineering environment”, SPIE, Edinburgh, Scotland, 2016
- Karban, R., “Using Executable SysML Models to Generate Systems Engineering Products”, NoMagic World Symposium, Allen, TX, 2016
- Open Source TMT model: https://github.com/Open-MBEE/TMT-SysML-Model
- Open Source Engineering Environment: https://open-mbee.github.io/
- Docgen, View&ViewPoints: https://github.com/Open-MBEE/mdk/tree/mdk-manual/src/main/dist/manual
- JPL Model-Based Systems Engineering Case Study: http://omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:incose_mbse_iw_2017:iw_2017_open_mbee.pdf
- A Practical Guide to SysML, 3rd Edition, Chapter 17 by Friedenthal, Moore, and Steiner
- Zwemer, D., “Connecting SysML with PLM/ALM, CAD, Simulation, Requirements, and Project Management Tools”, May 2016

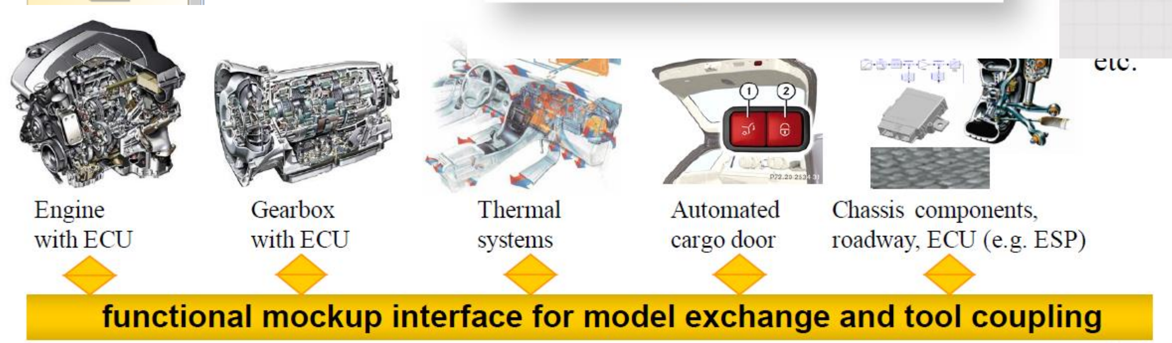
Outlook: Standardized Co-simulation

- The **Functional Mock-up Interface** (or **FMI**) defines a standardized interface to be used in computer simulations to develop complex cyber-physical systems.
- Integration with System Level behavior model

Tools supporting FMI	FMI Version	ModelExchange		CoSimulation		Notes
		Export	Import	Slave	Master	
Cameo Simulation Toolkit	FMI_1.0		Available		Available	FMUs can be imported, represented, connected and co-simulated in SysML models.

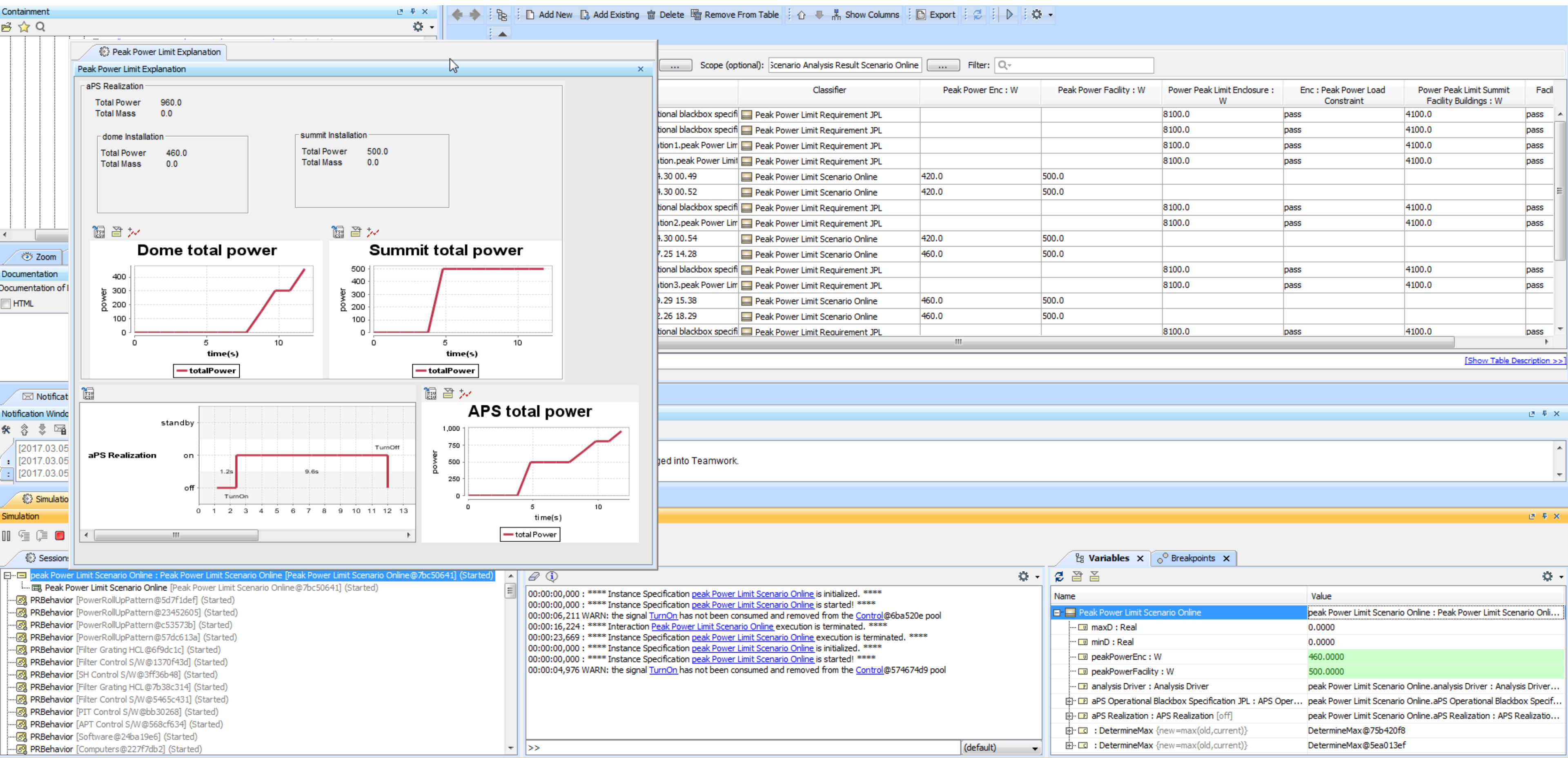


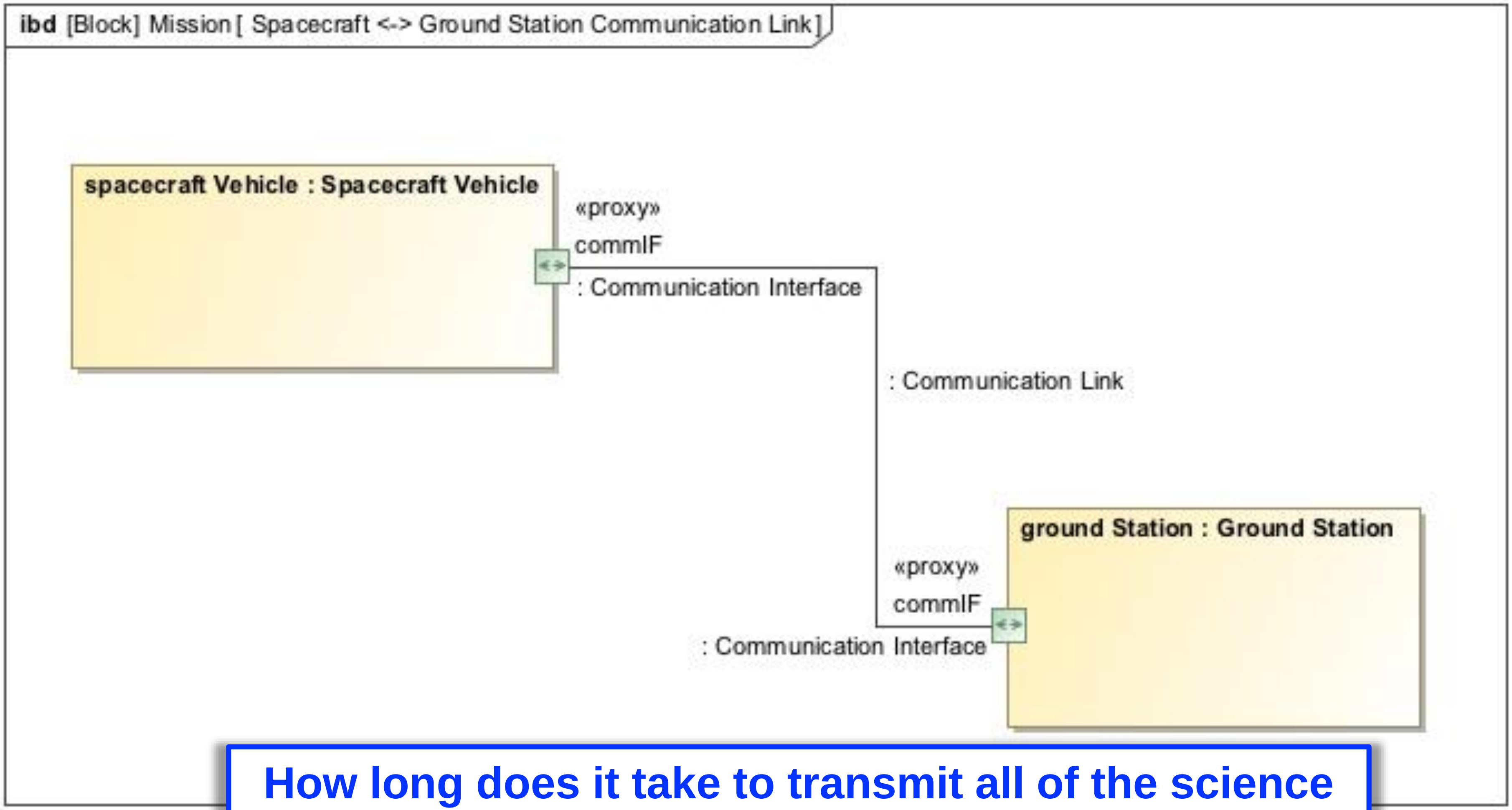




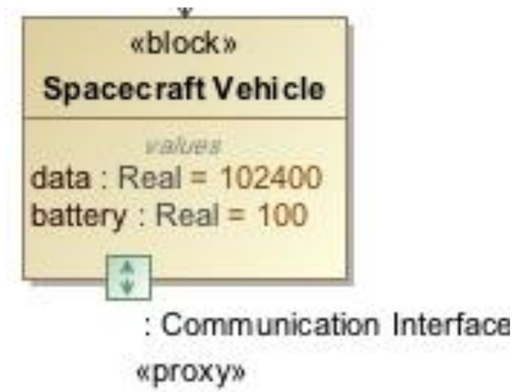
courtesy Daimler

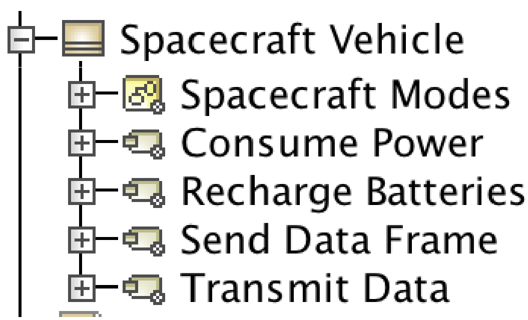
Power Analysis in the Context of Structure and Interfaces





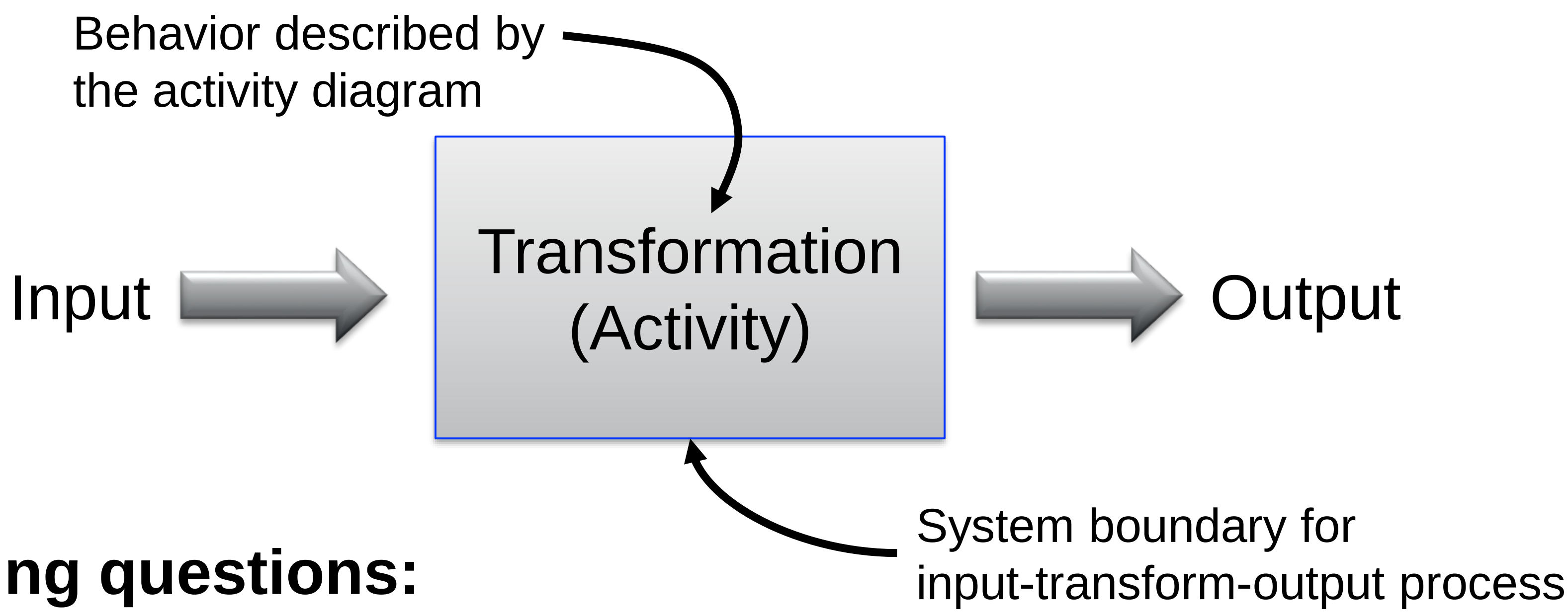
How long does it take to transmit all of the science data? What is the battery level profile over time?





- Specification of (owned) behavior of the object that is possibly invoked at some point in time
  - *Recharge Batteries*
  - *Transmit Data*
  - …

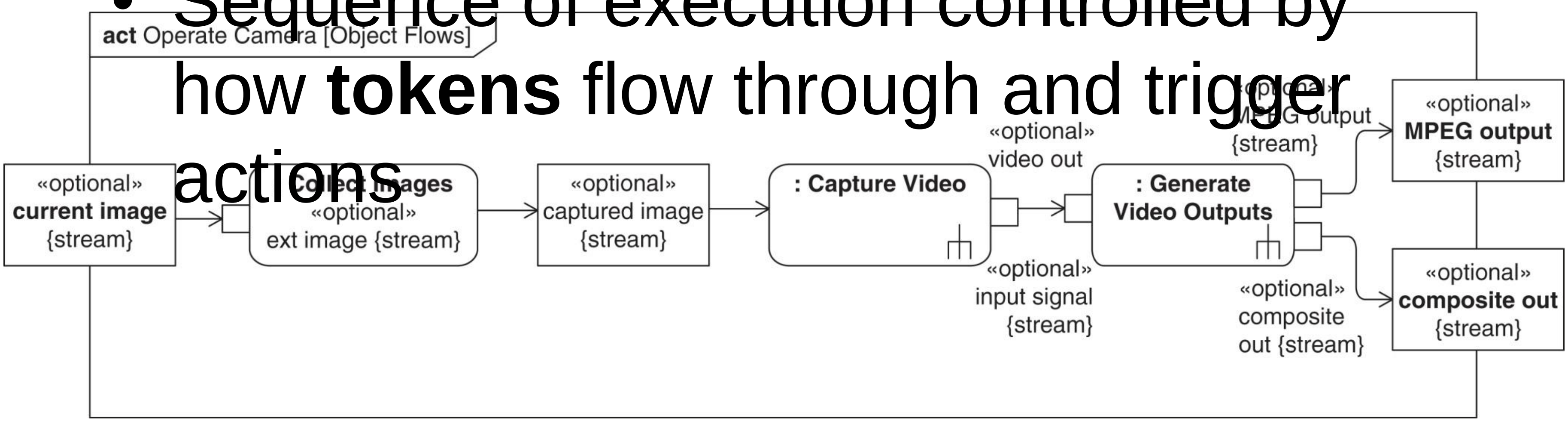
  ➔ Typically modeled using [activity diagrams]



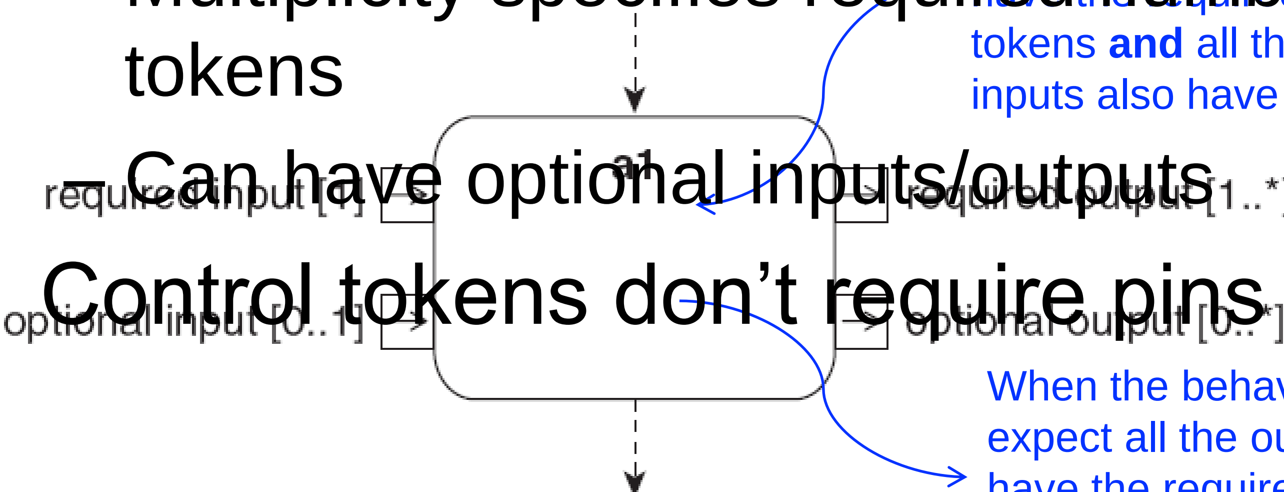
Modeling questions:

- What is the (system) boundary?
- What are the inputs & outputs?
- What are the object flows (things flowing through in/outs)?
- What is the transformation taking place?
- How is the transformation controlled?

- Specification of behavior through controlled sequence of actions
- An **activity** is decomposed into multiple actions with connecting **flows**
- Sequence of execution controlled by how **tokens** flow through and trigger actions

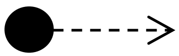


- Two types of flow: *object* and *control* flow
- Object tokens flow through pins
  - Multiplicity specifies required number of tokens
  - Can have optional inputs/outputs
- Control tokens don't require pins

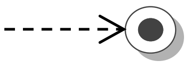


The behavior, a1, is not initiated until all the input pins have the required number of tokens **and** all the control inputs also have a token

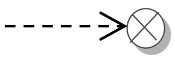
When the behavior ends, we expect all the output pins to have the required number of tokens **and** all the control outputs also get a token



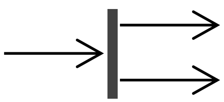
Initial Node – On execution of parent control token placed on outgoing control flows



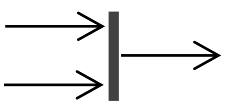
Activity Final Node – Receipt of a control token terminates parent



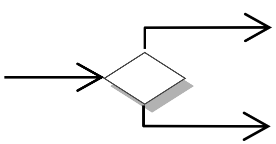
Flow Final Node – Sink for control tokens



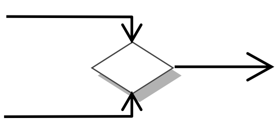
Fork Node – Duplicates input (control or object) tokens from its input flow onto all outgoing flows



Join Node – Waits for an input (control or object) token on <u>all</u> input flows and then places them all on the outgoing flow

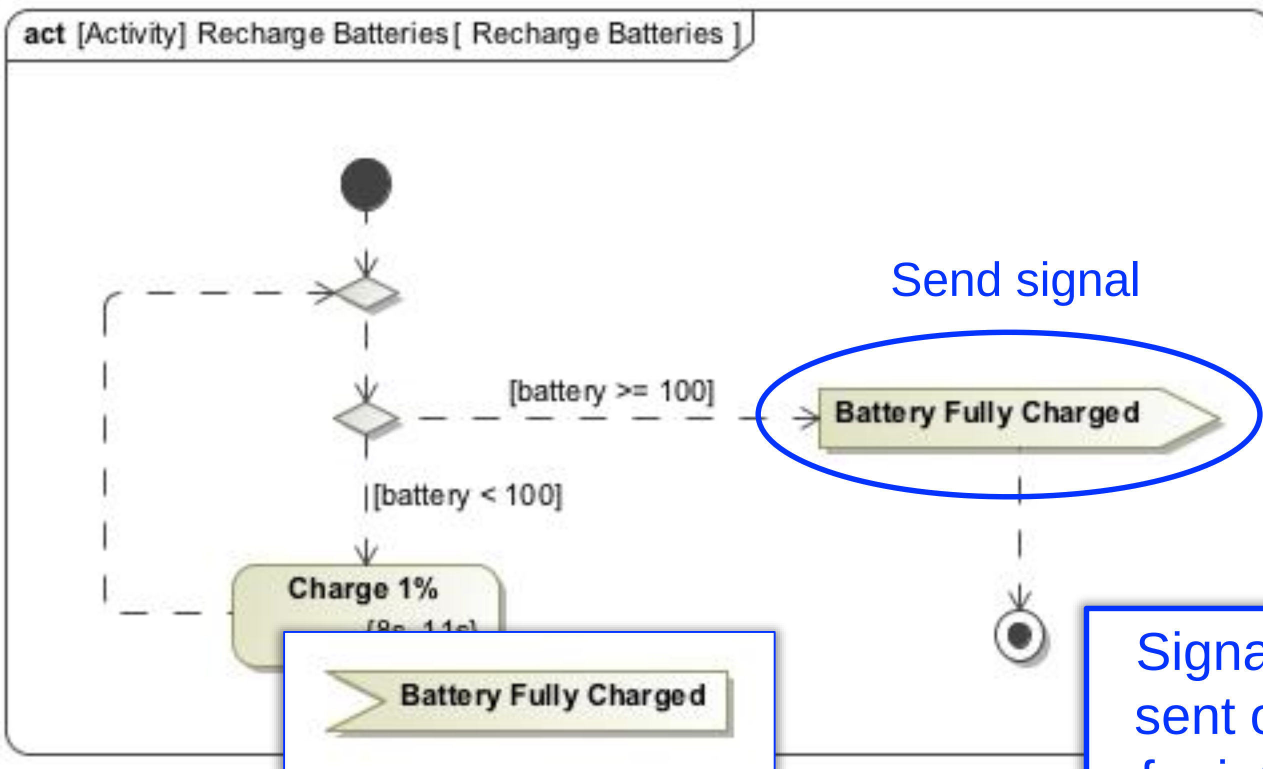


Decision Node – Waits for an input (control or object) token on its input flow and places it on one outgoing flow based on guards



Merge Node – Waits for an input (control or object) token on any input flows and then places it on the outgoing flow

Guard expressions can be applied on all flows

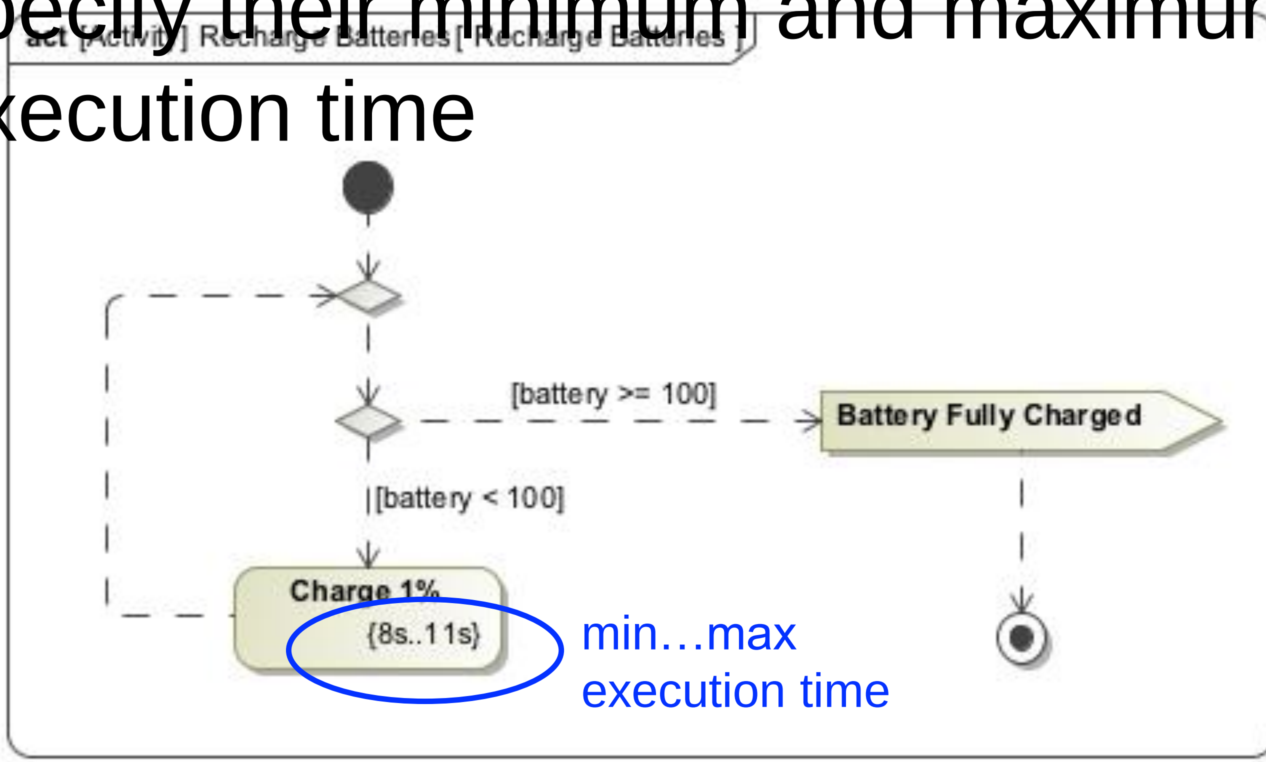


Send signal

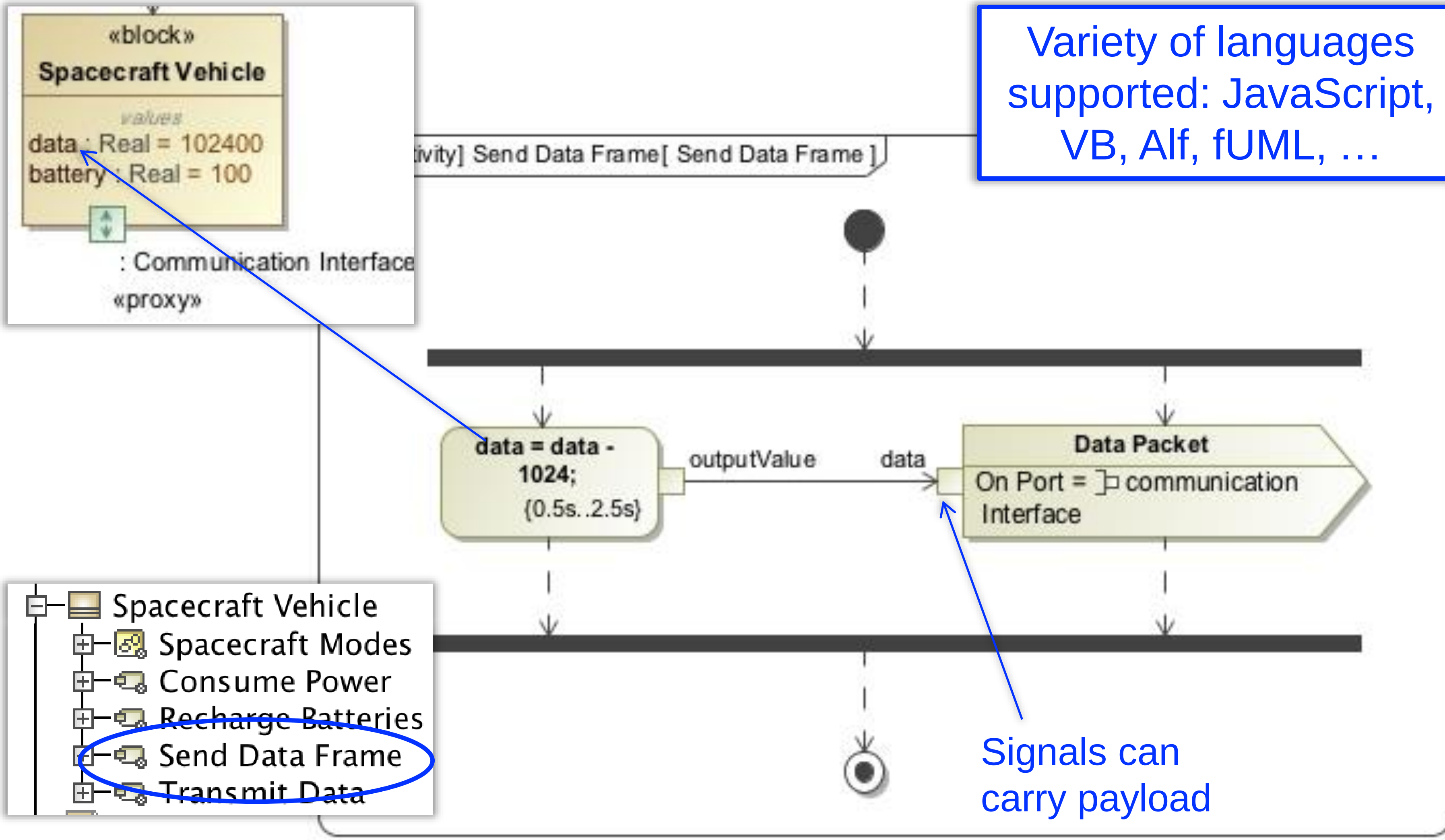
Wait for / receive signal (e.g., to trigger other behavior) *("Accept Event Action")*

Signals can be sent over ports for inter-object communication!

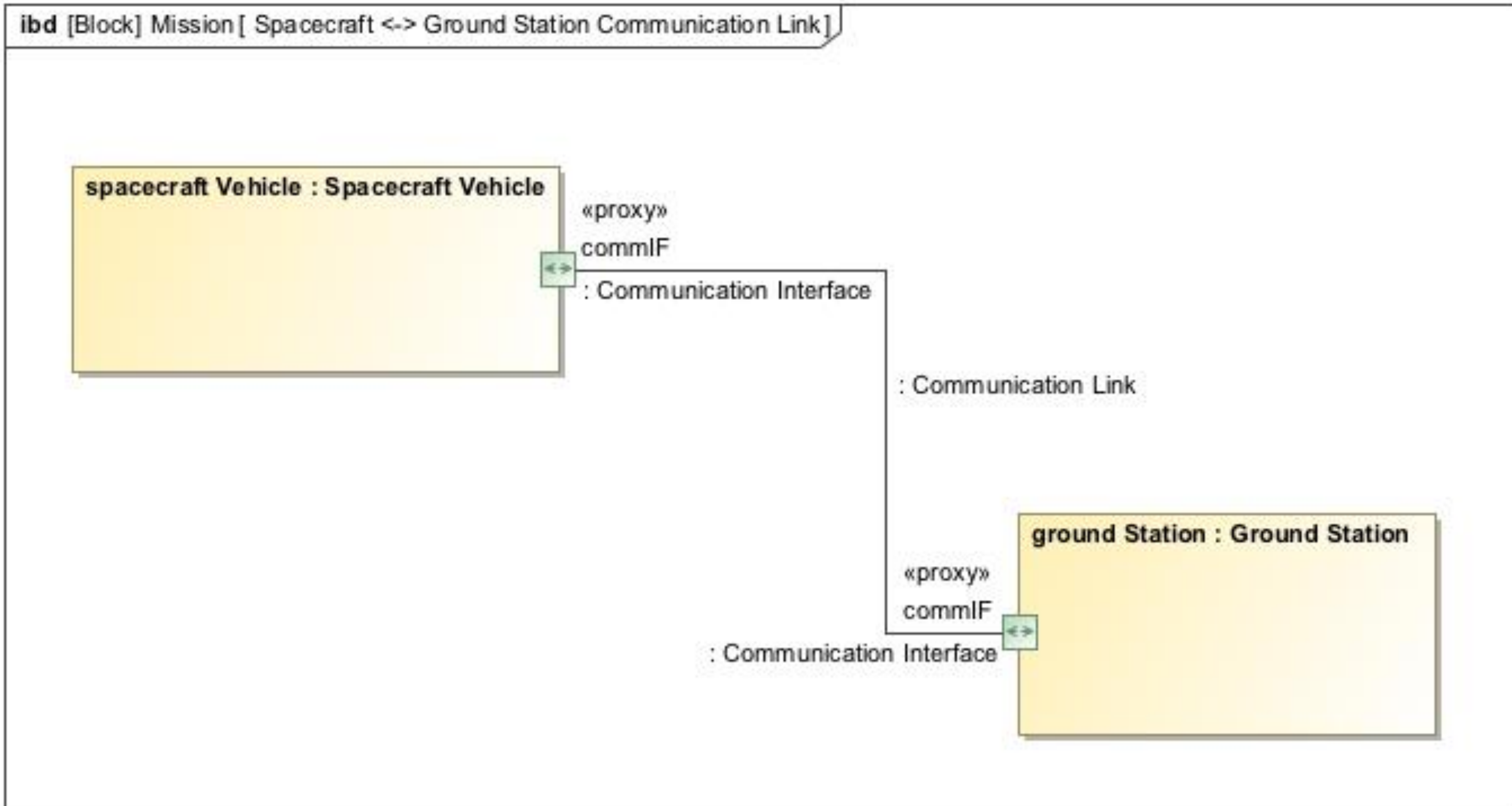
- Can apply "duration constraints" to **leaf** (non-decomposed) actions to specify their minimum and maximum execution time

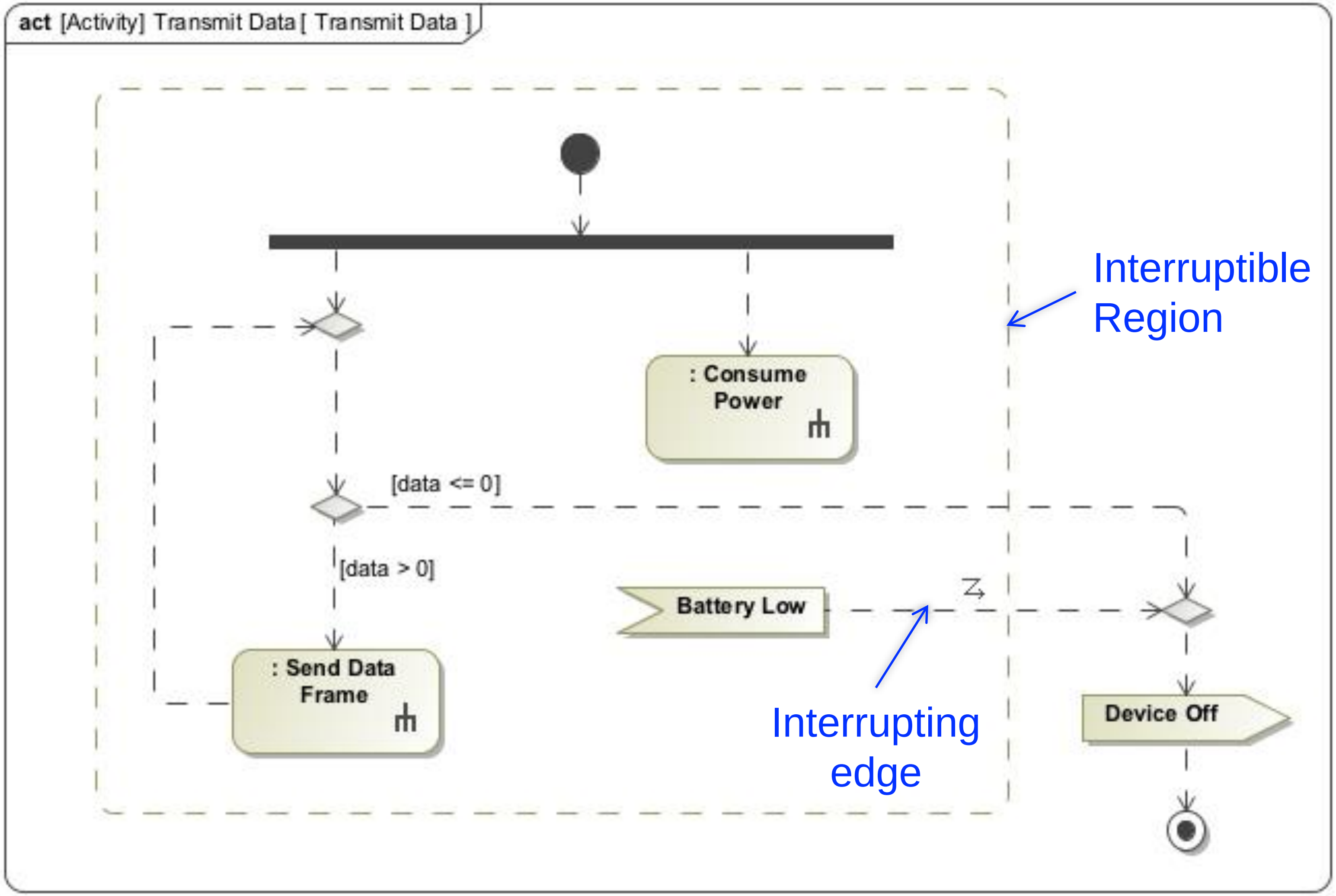


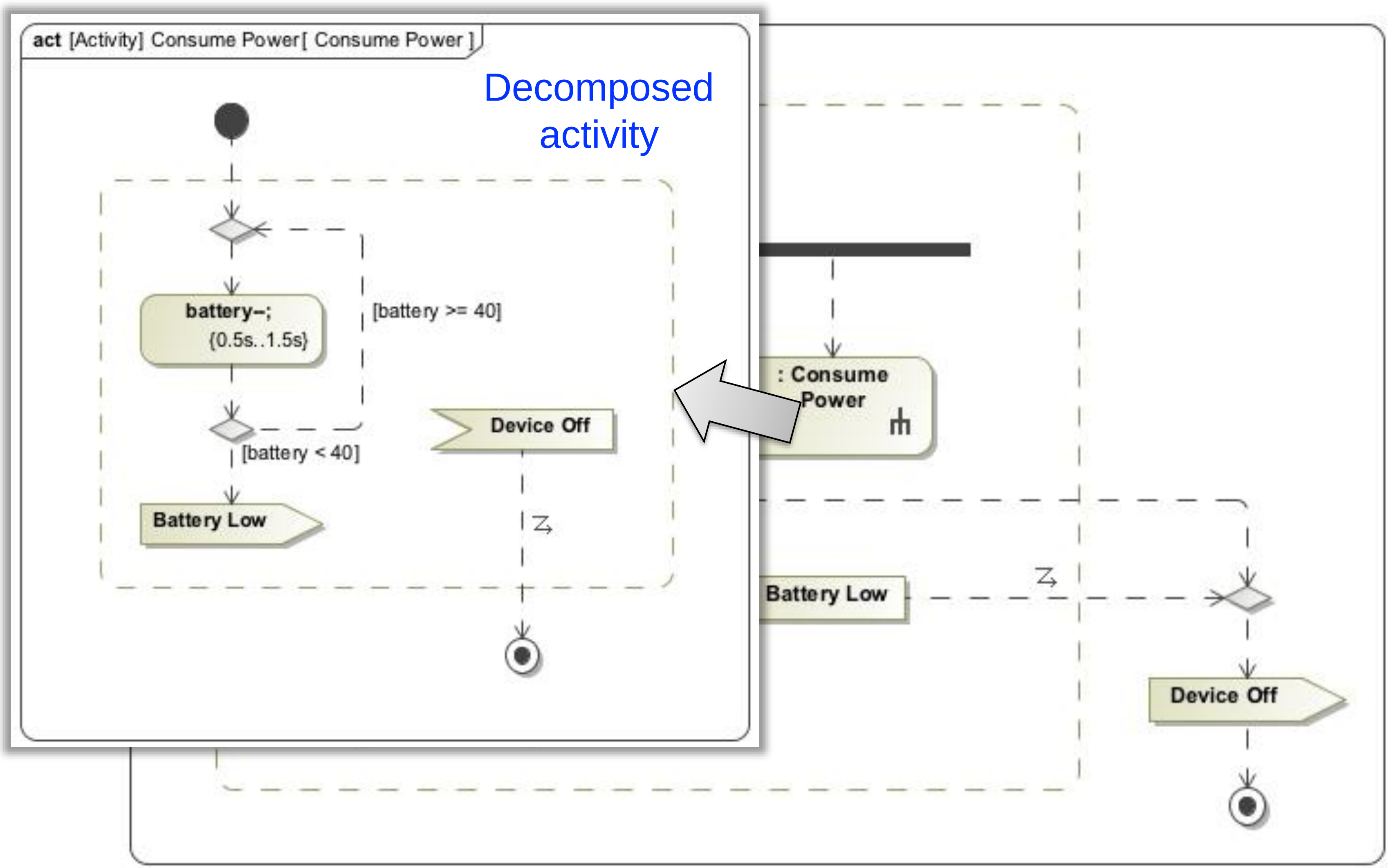
Variety of languages supported: JavaScript, VB, Alf, fUML, …



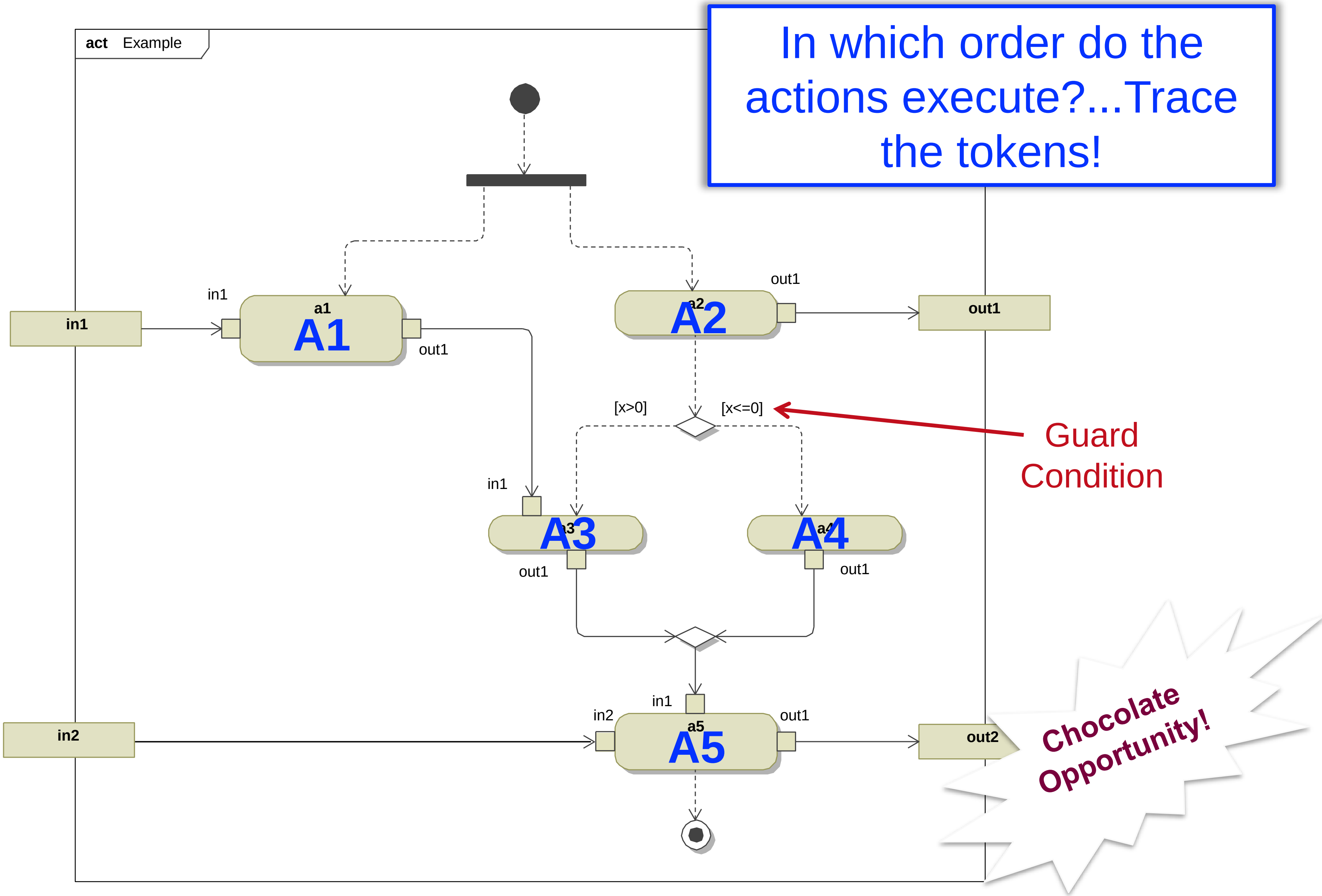
Signals can carry payload


ibd [Block] Mission [Spacecraft <-> Ground Station Communication Link]
spacecraft Vehicle : Spacecraft Vehicle
«proxy»
commIF
: Communication Interface
: Communication Link
ground Station : Ground Station
«proxy»
commIF
: Communication Interface

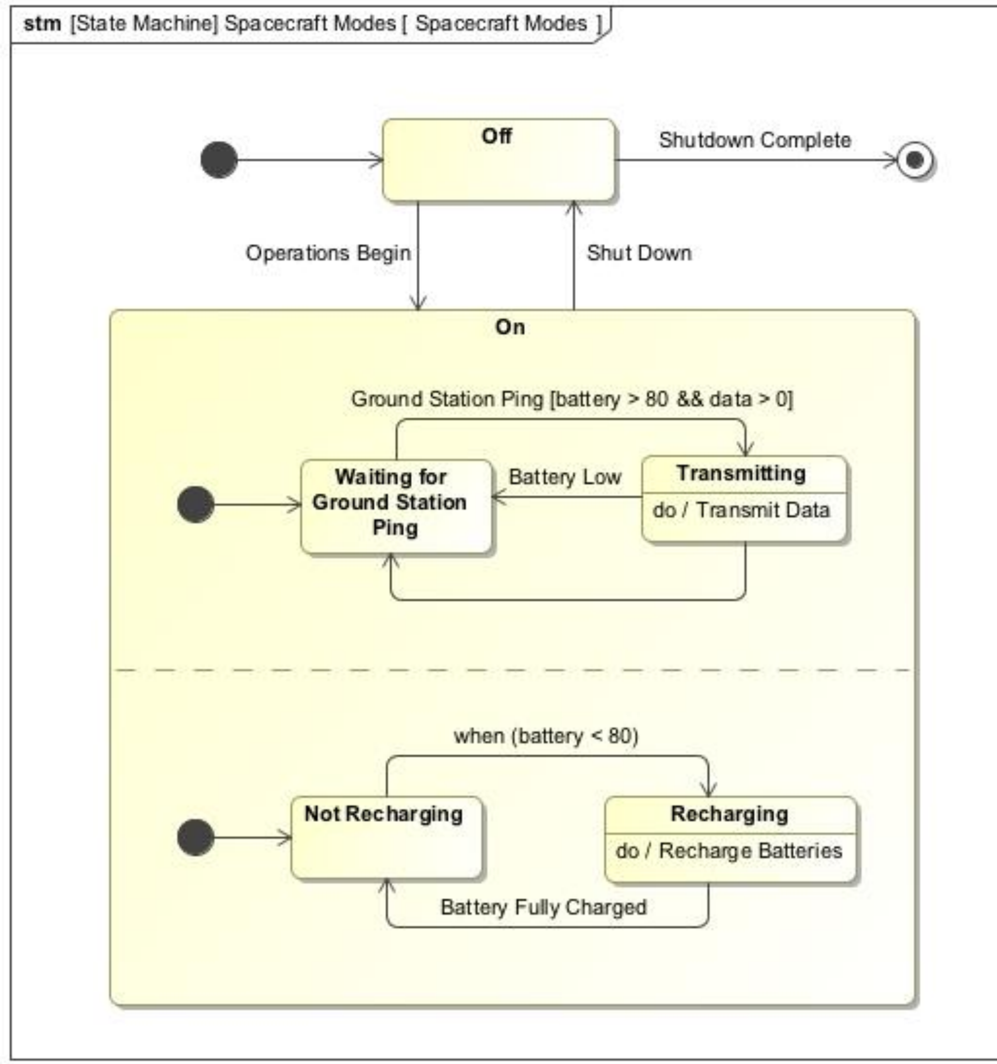

act [Activity] Transmit Data [Transmit Data]
: Consume Power
[data <= 0]
[data > 0]
Battery Low
: Send Data Frame
Device Off
Interrupting edge
Interruptible Region



act [Activity] Consume Power[Consume Power]
Decomposed activity
battery--;
{0.5s..1.5s}
[battery >= 40]
[battery < 40]
Device Off
Battery Low
: Consume Power
Battery Low
Device Off

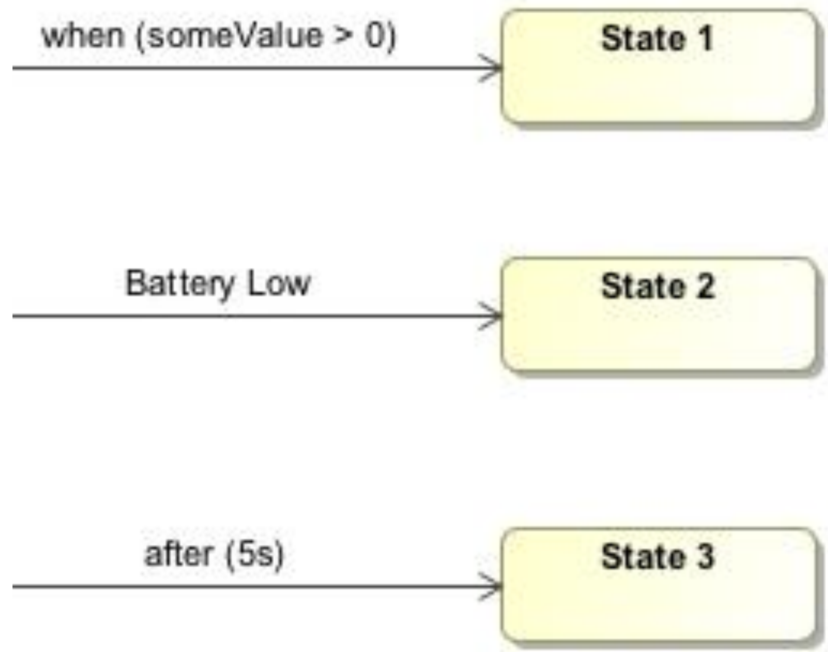

act Example
In which order do the actions execute?...Trace the tokens!
in1
in1
a1
A1
out1
a2
A2
out1
out1
[x>0]
[x<=0]
Guard Condition
in1
a3
A3
out1
a4
A4
out1
in2
in1
a5
A5
out1
in2
out2
Chocolate Opportunity!

- States (or "modes") and transitions between states
  - Transitions based on trigger and guard
  - Can send / receive signals to communicate between blocks

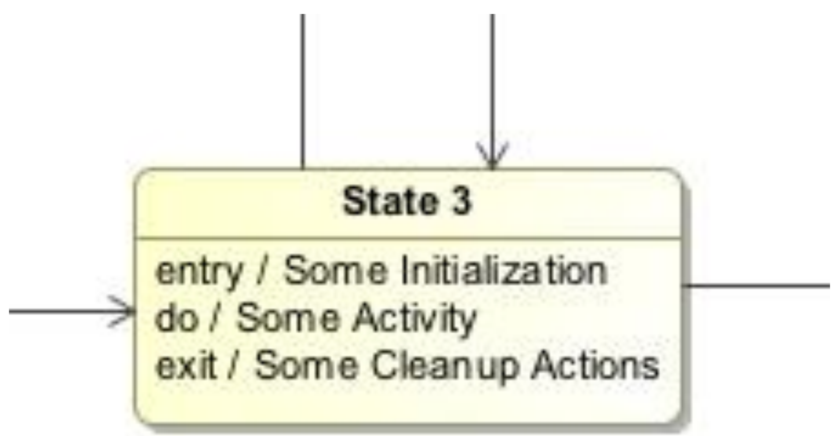


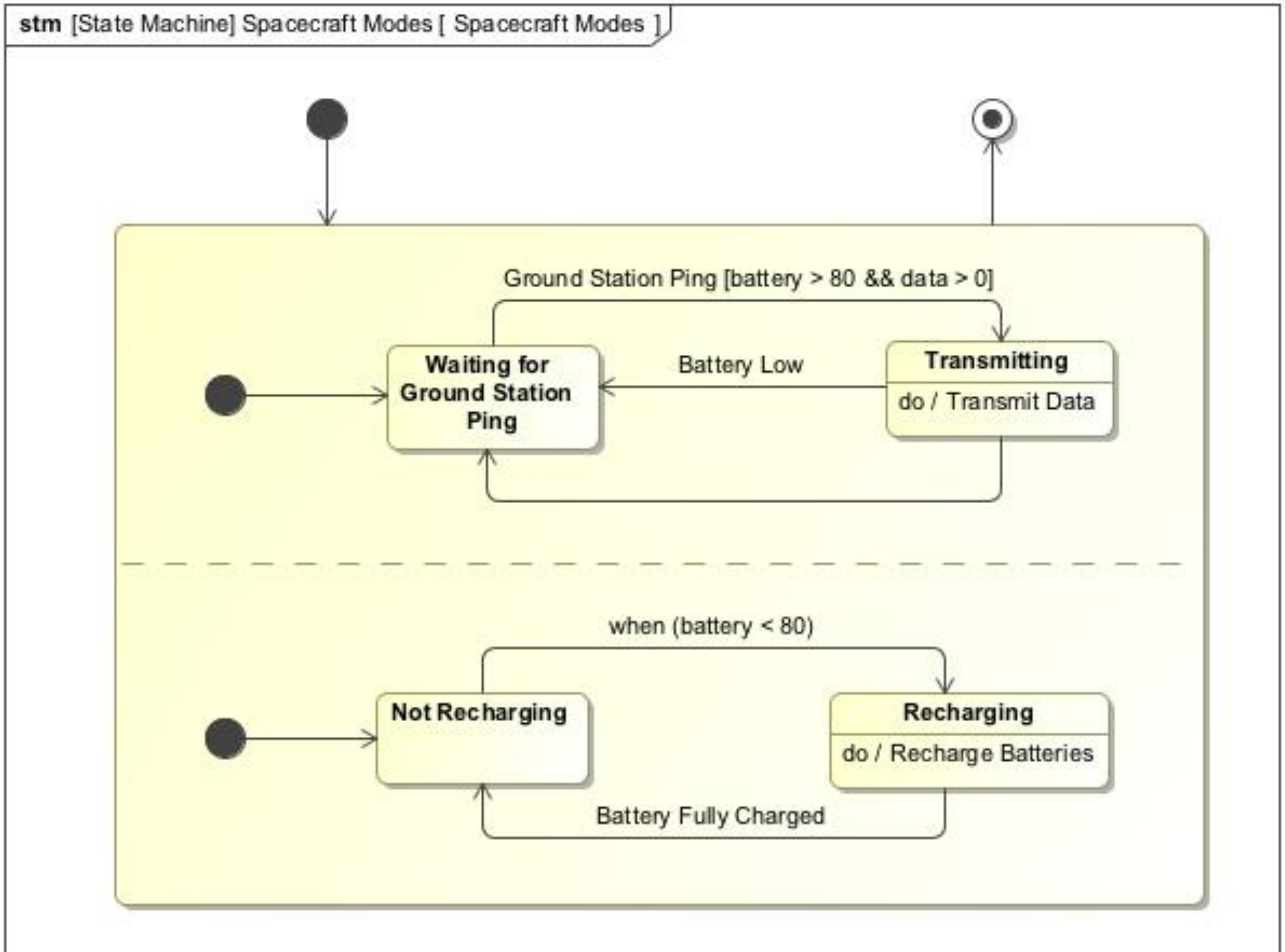
Typically used to

- Three major types of events for transitions:
  - **Change event**
  - **Signal event**
  - **Time event** (relative (*after*) or

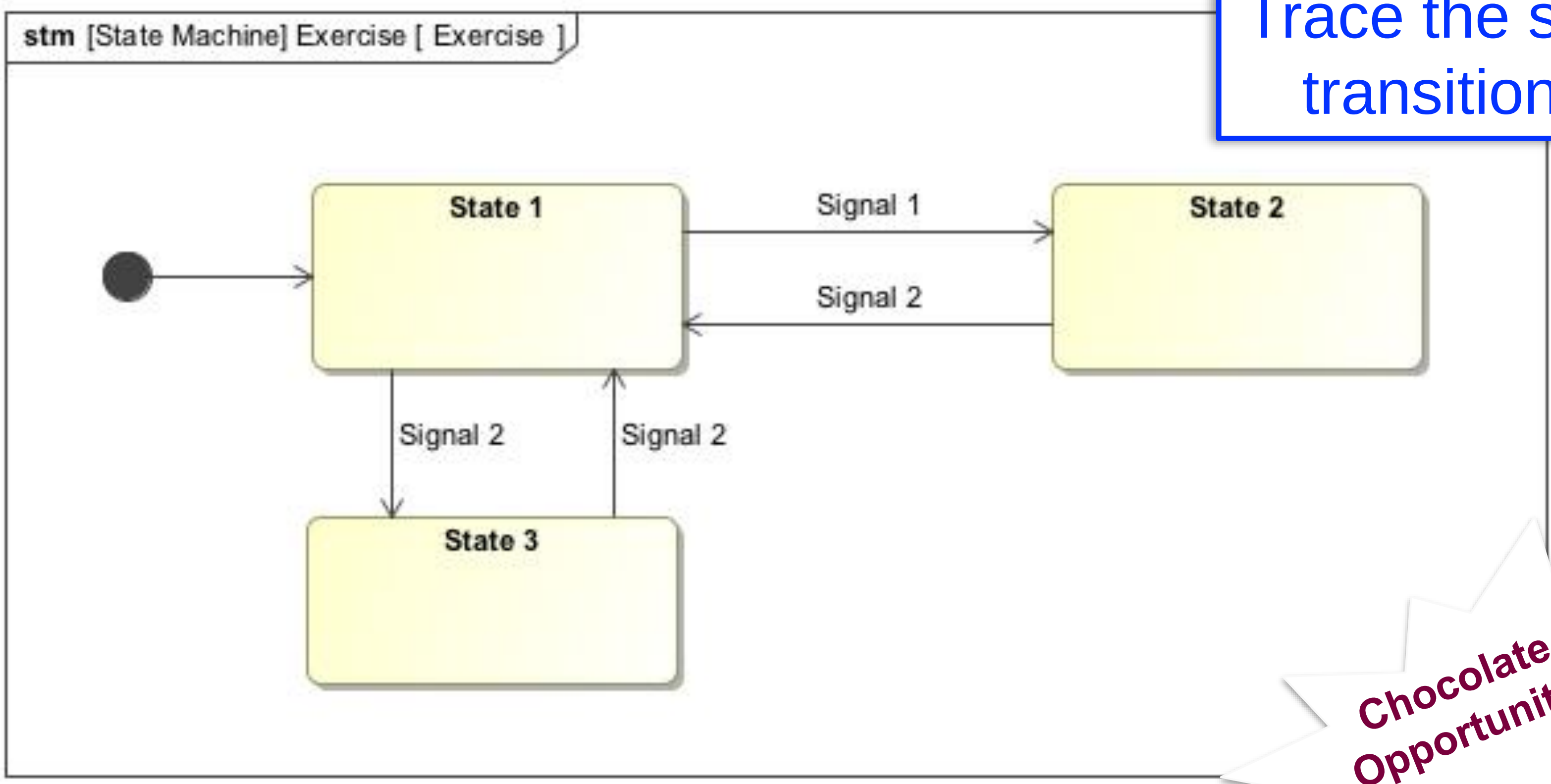


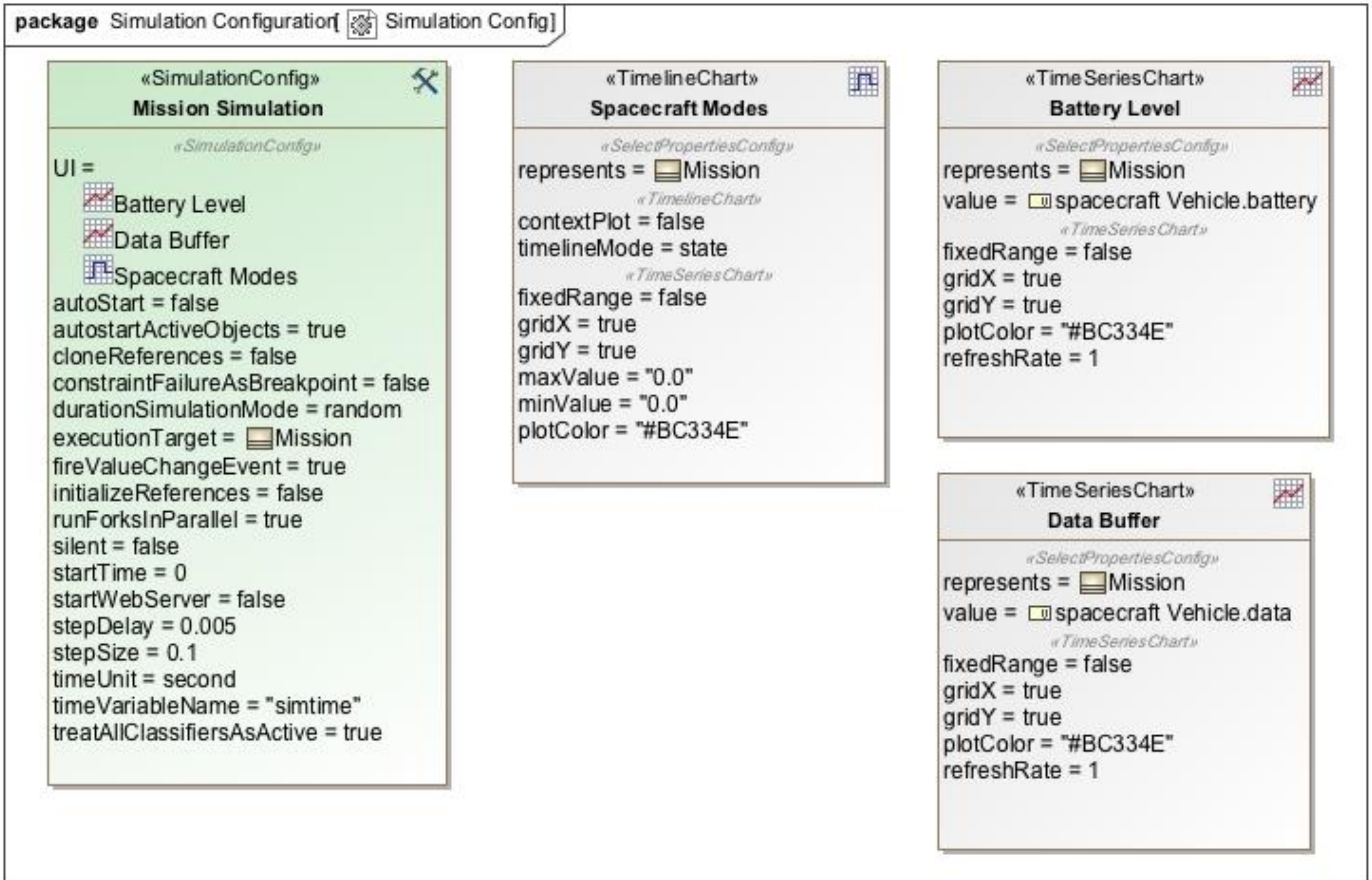
Transition notation:
trigger[guard]/action

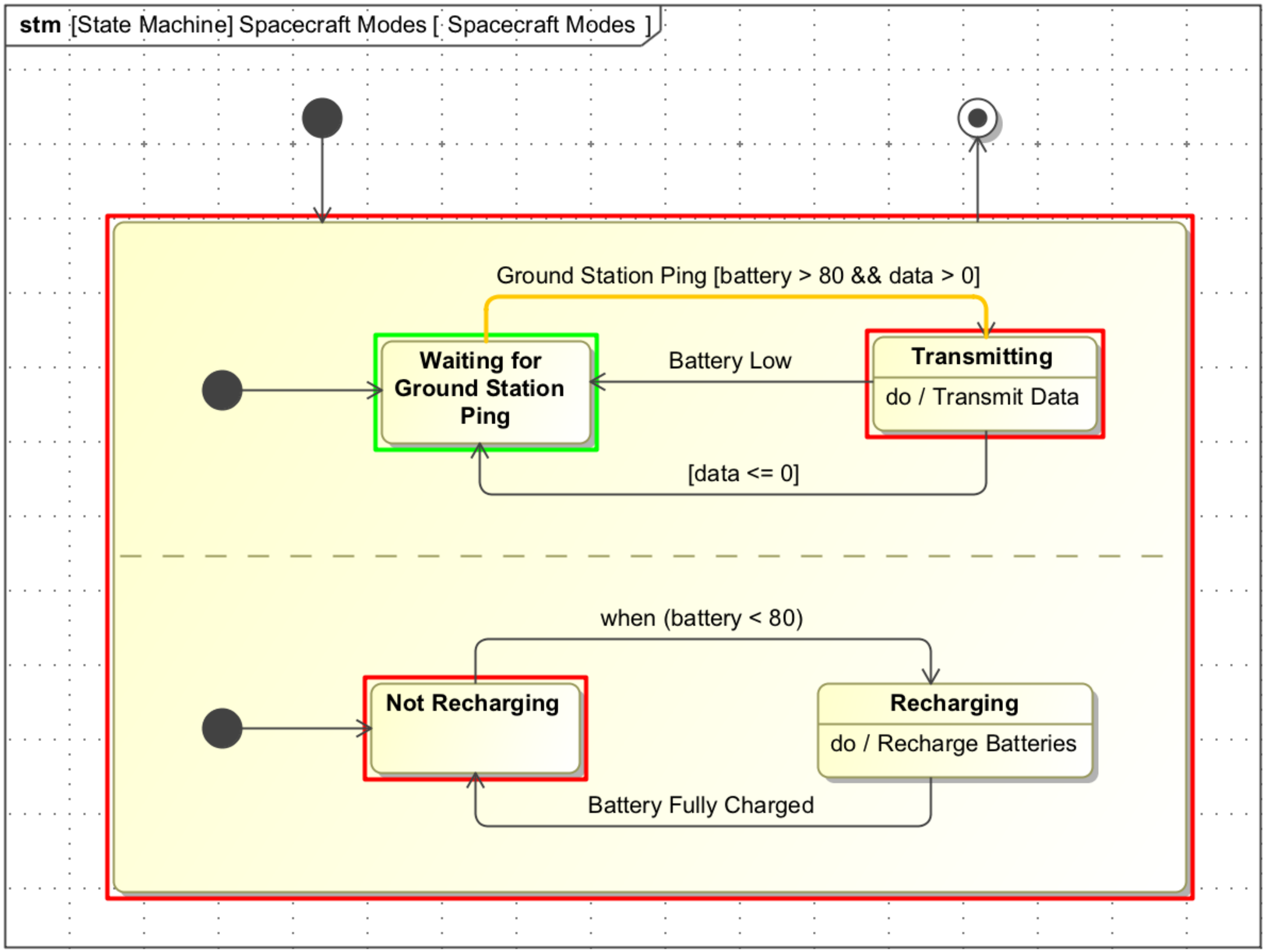

State 3
entry / Some Initialization
do / Some Activity
exit / Some Cleanup Actions

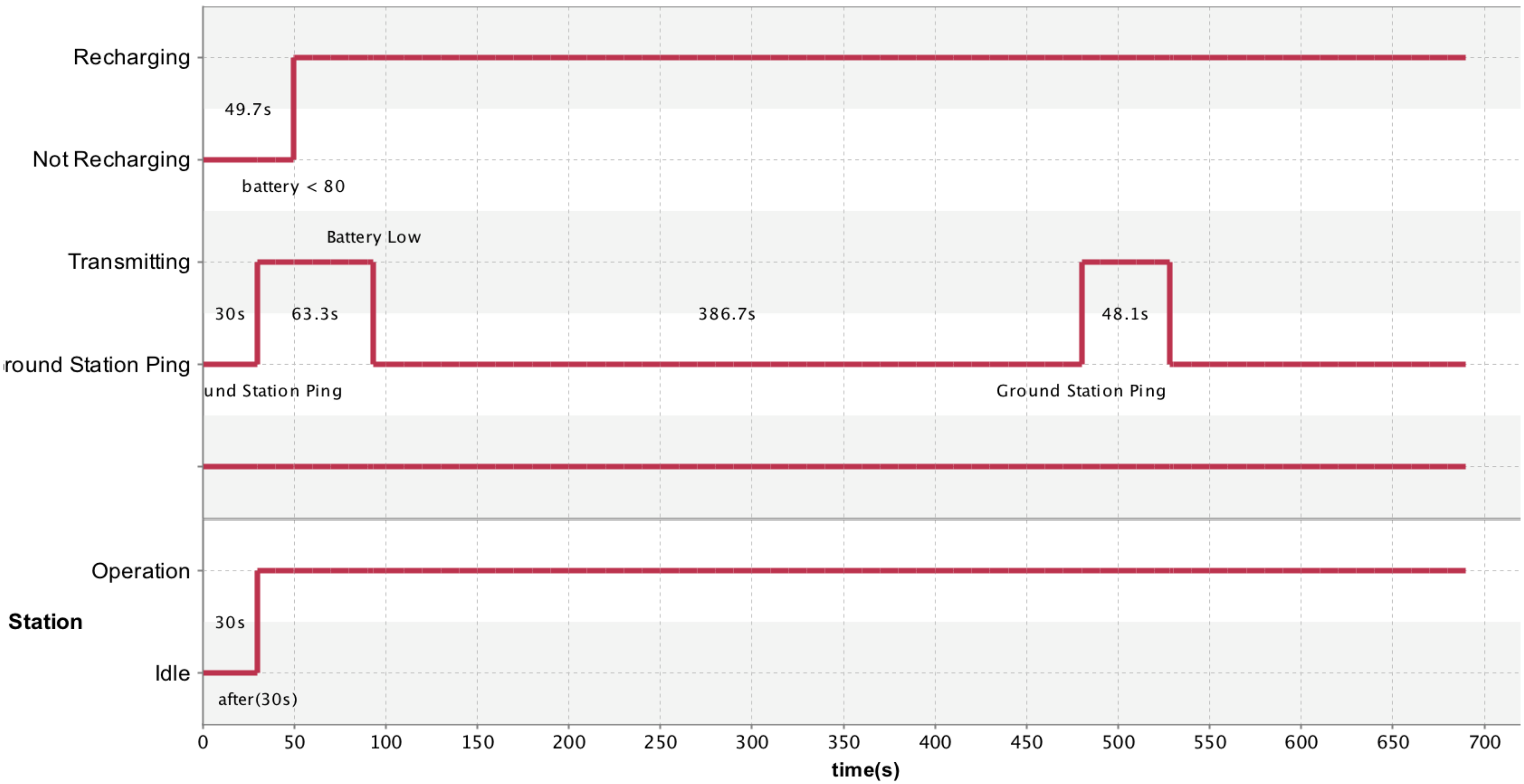

stm [State Machine] Spacecraft Modes [Spacecraft Modes]
Ground Station Ping [battery > 80 && data > 0]
Waiting for Ground Station Ping
Battery Low
Transmitting
do / Transmit Data
when (battery < 80)
Not Recharging
Recharging
do / Recharge Batteries
Battery Fully Charged

Trace the state transitions!

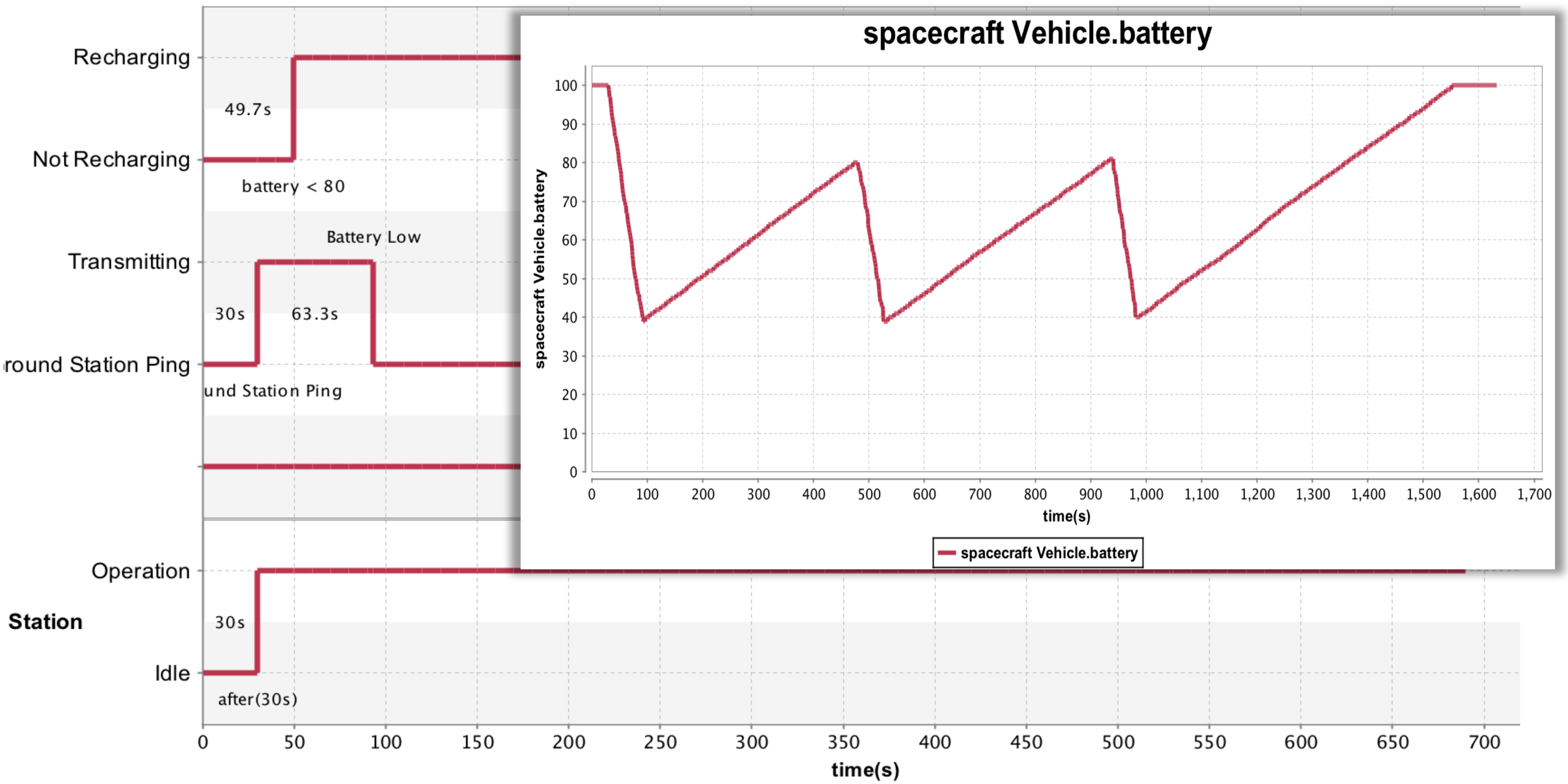



package Simulation Configuration[Simulation Config]
«SimulationConfig»
Mission Simulation
«SimulationConfig»
UI =
Battery Level
Data Buffer
Spacecraft Modes
autoStart = false
autostartActiveObjects = true
cloneReferences = false
constraintFailureAsBreakpoint = false
durationSimulationMode = random
executionTarget = Mission
fireValueChangeEvent = true
initializeReferences = false
runForksInParallel = true
silent = false
startTime = 0
startWebServer = false
stepDelay = 0.005
stepSize = 0.1
timeUnit = second
timeVariableName = "simtime"
treatAllClassifiersAsActive = true
«TimelineChart»
Spacecraft Modes
«SelectPropertiesConfig»
represents = Mission
«TimelineChart»
contextPlot = false
timelineMode = state
«TimeSeriesChart»
fixedRange = false
gridX = true
gridY = true
maxValue = "0.0"
minValue = "0.0"
plotColor = "#BC334E"
«TimeSeriesChart»
Battery Level
«SelectPropertiesConfig»
represents = Mission
value = spacecraft Vehicle.battery
«TimeSeriesChart»
fixedRange = false
gridX = true
gridY = true
plotColor = "#BC334E"
refreshRate = 1
«TimeSeriesChart»
Data Buffer
«SelectPropertiesConfig»
represents = Mission
value = spacecraft Vehicle.data
«TimeSeriesChart»
fixedRange = false
gridX = true
gridY = true
plotColor = "#BC334E"
refreshRate = 1


stm [State Machine] Spacecraft Modes [Spacecraft Modes]
Ground Station Ping [battery > 80 && data > 0]
Waiting for Ground Station Ping
Battery Low
Transmitting
do / Transmit Data
[data <= 0]
when (battery < 80)
Not Recharging
Recharging
do / Recharge Batteries
Battery Fully Charged



Recharging
Not Recharging
Transmitting
round Station Ping
Operation
Station
Idle
49.7s
battery < 80
Battery Low
30s
63.3s
386.7s
48.1s
und Station Ping
Ground Station Ping
30s
after(30s)
0
50
100
150
200
250
300
350
400
450
500
550
600
650
700
time(s)



spacecraft Vehicle.battery
spacecraft Vehicle.battery
time(s)
Recharging
Not Recharging
49.7s
battery < 80
Battery Low
Transmitting
30s
63.3s
round Station Ping
und Station Ping
Operation
Station
30s
Idle
after(30s)
time(s)

OpenCAE Environments and Technology Portfolio